

Ressource 1.03 - Bases de la programmation 1

Fascicule de travail : TP

Compétence ciblée	Traiter les données à des fins décisionnelles
AC couverts	AC11.05 Comprendre les structures algorithmiques de base et leur contexte d'usage – AC11.06 Prendre conscience de l'intérêt de la programmation
Description	<p>L'objectif de cette ressource est d'acquérir les principes de base de la programmation dans un langage de script. Contenus :</p> <ul style="list-style-type: none">– Structures de données : variables simples et structurées.– Structures de contrôles : alternatives et boucles.– Lecture et écriture de fichiers. <p>On profitera des enseignements pour présenter les bonnes pratiques dans la programmation (nommage des variables et des fonctions, indentation, syntaxe, tests, commentaires, documentations...)</p> <p>Cette ressource présente les éléments, concepts et principes de base de la programmation dont la maîtrise est indispensable dans l'acquisition de la compétence.</p>
Mots clés	Programmation – langage
Heures formation	30h
Contrôle Ecrit	1h30 (coefficient 50%)
Contrôle Pratique	1h (coefficient 50%)
Semestre	Semestre 1
Vos contacts	Mohammed Azerkane , chargé de TPs : azerkane.m@gmail.com Dorine Tabary , responsable de l'UE : dorine.tabary@univ-fcomte.fr

Organisation des séances de TP

semaine 39	Premiers programmes
semaine 40	Les structures de contrôle
semaine 41	Lecture/ écriture de fichier
semaine 42	Exemple de sujet de contrôle

Examen des connaissances pratiques

Comment se passera l'examen ? L'examen sera tiré de sujets d'advent of code^a.
Seront pris en compte les commentaires, et le respect des règles de bon codage.

^a<https://adventofcode.com/>

Les Travaux Pratiques

L'objectif est d'acquérir les notions de base de la programmation Python.

Des corrections sont disponibles sur le git.

Il est donc contre productif de copier directement les solutions sur internet ou d'utiliser un LLM (vous aurez tout le loisir les années suivantes d'en manipuler).

Lors de chaque TP, les premiers exercices non optionnels doivent être réalisés. La partie optionnelle concerne les plus aguerris d'entre vous. Pour ces derniers, le jeu est d'aller le plus loin possible dans le temps imparti.

Il est également obligatoire à la fin de chaque TP, de compléter la partie BILAN.

Evénements

Il existe plusieurs événements informatiques.

Nuit de l'info. **Date :** 7 et 8 décembre 2023, de 16h38 à 08h06. En présentiel.

Détails : Un serious-game regroupant des milliers d'étudiants pour développer une application informatique en une nuit. L'événement a lieu dans l'université de Besançon (route de gray, bâtiment C) et se déroule par équipe.

Contact : Mme Tabary (membre de l'équipe organisatrice)

Global Game Jam. **Date :** mi-janvier 2024. En présentiel avec possibilité de distanciel.

Détails : L'événement se déroule en 48 heures, pendant lesquelles les développeurs doivent concevoir un jeu vidéo à partir d'un thème partagé par tous les participants. Leurs créations sont ensuite qualifiées par un jury. L'événement a lieu dans l'université de Besançon (maison des étudiants) et se déroule par équipe.

Contact : Mme Tabary (participante, ou membre de l'équipe organisatrice si pénurie)

Advent of code. **Date :** 1 - 25 décembre 2023. En distanciel.

Détails : L'Advent of Code (<https://adventofcode.com/>) est une série annuelle de défis de programmation informatique qui suit un calendrier de l'Avent.

Les énigmes (puzzles) de programmation couvrent un vaste champ de compétences et peuvent être résolues en utilisant n'importe quel langage de programmation. Les participants s'affrontent également en fonction de leur vitesse dans les classements mondiaux et privés. En individuel.

Contact : Mme Tabary (participante), avec possibilité de s'incruster dans un groupe privé.

Le manuel

<https://docs.python.org/fr/3/contents.html>

Contents

1	TP1 : Premiers programmes	3
1.1	Installer et configurer Python	3
1.2	Rangement de votre espace de travail	3
1.3	Echauffement	3
1.4	Le juste prix	8
1.5	Bilan (obligatoire)	9
2	TP2 : Les structures de données	10
2.1	Lists	10
2.2	Tuples	13
2.3	Sets	15
2.4	Dictionnaires	17
2.5	Projets (optionnels)	18
2.6	Bilan (obligatoire)	19
3	TP3 : Lecture / écriture de fichier	20
3.1	Fichiers de type texte (.txt)	20
3.2	Les fichiers csv	21
3.3	Fichiers de type xlsc	23
3.4	Fichiers de type json	25
3.5	Bilan (obligatoire)	27
4	TP4 : Entraînement contrôle	28
4.1	Exercice 1. Année 2015, jour 1	28
4.2	Exercice 2. Année 2015, jour 2	28
4.3	Exercice 3. Année 2015, jour 3	29
4.4	Exercice 4. Année 2015, jour 5	30
5	Corrections	32
5.1	TP 1	32
5.2	TP 2	34
5.3	TP 3	38
5.4	TP 4	39

1 TP1 : Premiers programmes

1.1 Installer et configurer Python

(Inutile avec les ordinateurs de l'université, déjà configurés).

Python est un **langage interprété**. Contrairement au langage compilé qui fournit un code binaire utilisable et réutilisable par la machine, le langage interprété nécessite d'utiliser un interpréteur à chaque exécution.

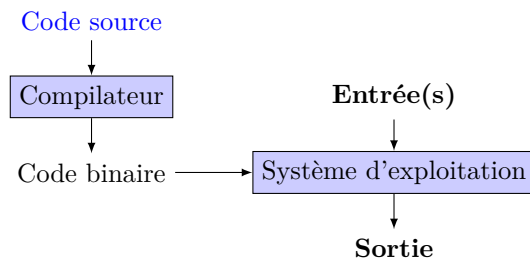


Figure 1: Langage compilé

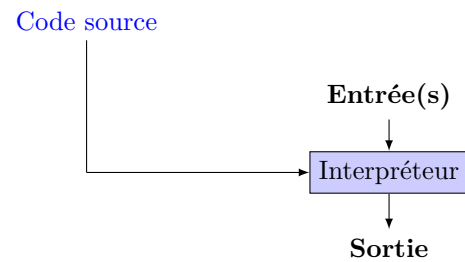


Figure 2: Langage interprété

Avant l'installation, n'oubliez pas de vérifier que Python est bien présent de votre ordinateur.

Une aide à l'installation est disponible sous ce lien : <https://www.commentcoder.com/installer-python/>

Vos professeurs peuvent également vous aider pour installer l'environnement de travail sur votre machine personnelle (très recommandé).

1.2 Rangement de votre espace de travail

Tous vos programmes devront être rangés dans des dossiers. Par exemple, enregistrez votre programme dans un dossier Cours, dans un dossier Semestre 1, dans un dossier Bases de la programmation, dans un dossier tp1.

1.3 Echauffement

1.3.1 Ecrire et lire sur un terminal de commande

Sortie écran Pour afficher une chaîne de caractères en Python 3, ouvrez un éditeur de texte et écrivez :

```
1 print("premier programme");
```

- Enregistrez ce dossier sous le nom de premier.py
- Dans le terminal, exécuter : `python3 premier.py`

Avec le terminal de commande, allez dans le dossier contenant votre programme.

```

plproadmin@pdept16:~$ cd Bureau/Cours/sd/ue/
r101bp/ saé1.01/ saé102/ saé1.04/ semio/
plproadmin@pdept16:~$ cd Bureau/Cours/sd/ue/r101bp
plproadmin@pdept16:~/Bureau/Cours/sd/ue/r101bp$ ls
02.initiation.pdf latexPres premier.py
  
```

Figure 3: Commandes pour se déplacer dans la hiérarchie de fichiers.

Exécutez le programme avec l'instruction `python3 premier.py`.

```
plproadmin@pdept16:~/Bureau/Cours/sd/ue/r101bp$ python3 premier.py
premier programme
```

Figure 4: Commande pour exécuter un programme.

Exercices

A partir du programme `premier.py`, écrivez les programmes suivant : Exercice

1♦- Ecrire le programme qui ne prend rien en entrée et donne en sortie le texte suivant.

```
plproadmin@pdept16:~/Bureau/Cours/sd/ue/r101bp$ python3 second.py
Hello
ceci est un programme
en python !
```

2♦- Sauriez-vous le faire en n'utilisant qu'une seule fois l'instruction `print` ?

Entrée utilisateur Pour entrer une chaîne de caractères en Python 3, ouvrez un éditeur de texte et écrivez :

```
1 """
2 Définition: Programme qui demande à l'utilisateur son nom et affiche ce nom sur le terminal.
3 réalisé dans le cadre du module R1.03 Bases de la programmation
4 Auteur: Mme Tabary
5 Date: 2/08/2023 Version: 1.0
6 """
7
8 #affichage sur le terminal
9 print("Bonjour, quel est ton nom ?")
10 #saisir une valeur au clavier avec input et affectation de la variable nom
11 nom = input()
12
13 # affichage du contenu de la variable nom
14 print("Bonjour humain au nom de", nom, ".")
```

La sortie écran obtenue est la suivante :

```
plproadmin@pdept16:~/Bureau/Cours/sd/ue/r101bp$ python3 programmeAvecEntrees.py
Bonjour, quel est ton nom ?
Tabary
Bonjour humain au nom de Tabary .
```

Tip for Code 1 : *"code is read much more often than it is written"*, Guido van Rossum

Commentez un paragraphe avec : `"""`.

Commentez une ligne avec : `#`.

Et quand vous codez, rajoutez des commentaires.

<https://peps.python.org/pep-0008/>

Exercices

1◇- Ecrire le programme qui donne en sortie le texte suivant.

```
Bonjour, quel est ton nom ?
Tabary
quel est ton âge ?
34
Bonjour humain au nom de Tabary et âgé de 34 ans.
```

2◇ ◇- Modifiez le programme afin d'afficher en plus l'année de naissance.

```
Bonjour, quel est ton nom ?
Tabary
quel est ton âge ?
34
Bonjour humain au nom de Tabary , âgé de 34 ans et né en 1989 .
```

Aide : Tip for Code 2

3◇ ◇ ◇ (**optionnel**)- On suppose en plus que la taille est demandée. L'utilisateur entre un texte qui peut être selon les formats suivants : 177.56cm, ou 177cm, ou 17756cm. Le terminal répond dans tous les cas :

```
Bonjour, quel est ton nom ?
Tabary
quel est ton âge ?
34
quel est ta taille ?
177.56cm
Bonjour humain au nom de Tabary , âgé de 34 ans, né en 1989 et mesurant 1 m 77 cm.
```

Aide : Tip for Code 2 (encore)

Tip for Code 2 : "Le shell est votre ami"

Les bugs, ou erreurs de programmation, sont inévitables dans le développement de logiciels complexes, même pour les programmeurs les plus expérimentés. Il faut savoir lire le code erreur associé.

Une erreur qui risque d'apparaître dans votre cas est la suivante :

```
Traceback (most recent call last):
  File "entree2.py", line 20, in <module>
    print("Bonjour humain au nom de", nom, ", âgé de ", age, " ans et né en ", A
NNEE - age, ".")
TypeError: unsupported operand type(s) for -: 'int' and 'str'
```

Dans ce cas, l'erreur se situe **ligne 20**.

Une variable de type string (texte) est utilisée à la place d'une variable de type int (entier).

En effet ANNEE est une variable globale de type entier (int). Or, il est impossible de soustraire un texte d'un entier. Il faut donc changer le type de la variable age afin qu'elle soit un entier. Cette démarche s'appelle le transtypage. Pour ce faire, vous pouvez écrire : (int)age.

Source : <https://docs.python.org/fr/3/tutorial/inputoutput.html>

1.3.2 Les conditionnelles

Maintenant, l'objectif est de maîtriser l'alternative¹. Testez le programme suivant en respectant bien les indentations.

```

1 """
2 Définition: Programme qui demande à l'utilisateur son nom et affiche ce nom sur le terminal.
3 réalisé dans le cadre du module R1.03 Bases de la programmation
4 Auteur: Mme Tabary
5 Date: 3/08/2023    Version: 1.0
6 """
7
8 #affichage sur le terminal
9 print("quel est ton âge ?")
10 #saisir une valeur au clavier avec input et affectation de la variable age
11 age = int(input())
12
13 # affichage du contenu de la variable nom
14 print("Bonjour humain âgé de ", age, "ans.")
15
16 # test de la valeur de la variable age
17 if age < 18:
18     print("Tu es mineur")
19 else:
20     print("Tu es majeur")

```

Tip for Code 3 : Indentations et espaces

Les annotations pour les variables doivent comporter un seul espace après les deux points. Il ne doit pas y avoir d'espace avant les deux points (voir ligne 17). Si une affectation a un côté droit, le signe d'égalité doit avoir exactement un espace des deux côtés (voir ligne 11).

Source : Variable Annotations de <https://peps.python.org/pep-0008/>
la commande `typeof(maVariable)` permet de récupérer le type de la variable.

Exercices

- 1◇- Ecrire le programme qui spécifie si la personne est mineur, majeure, ou est dans l'année de sa majorité.
- 2◇- Modifier le programme pour incorporer une variable état qui prend la valeur "mineur", "majeure", ou "est dans l'année de sa majorité", selon l'âge de entrée par l'utilisateur.

1.3.3 Les itératives

boucle for Maintenant, l'objectif est de travailler la boucle `for`². Testez le programme suivant en respectant bien les indentations.

```

1 """
2 Définition: Programme qui affiche les éléments d'une liste.
3 réalisé dans le cadre du module R1.03 Bases de la programmation
4 Auteur: Mme Tabary
5 Date: 3/08/2023    Version: 1.0

```

¹<https://docs.python.org/fr/3/tutorial/controlflow.html#if-statements>

²https://docs.python.org/fr/3/reference/compound_stmts.html#the-while-statement

```

6 """
7
8 #tableau de la classe
9 listeEtudiants = ["Alice", "Bob", "Charlie", "Diane"]
10
11 #verification de la présence des étudiants
12 for w in listeEtudiants:
13     print("nom de l'étudiant : ",w)

```

Exercices

1◇- Modifier le programme pour incorporer la question pour chaque étudiant "L'étudiant est-il venu en cours ? (n/y)". Si la réponse est "n", alors le prénom de l'étudiant est rajouté à une liste d'étudiants absents. Ensuite afficher la nouvelle liste des étudiants absents.

```

L'étudiant Alice est-il venu en cours ?
y
L'étudiant Bob est-il venu en cours ?
n
L'étudiant Charlie est-il venu en cours ?
n
L'étudiant Diane est-il venu en cours ?
y
Nom de l'étudiant absent : Bob
Nom de l'étudiant absent : Charlie

```

(optionnel) Quel est, selon la taille de la liste d'entrée, le nombre d'opérations du programme ?

(optionnel) 2◇ ◇- Réaliser un programme vérifiant qu'aucun étudiant n'est en double dans le tableau.

Quel est, selon la taille de la liste d'entrée, le nombre d'opérations du programme ?

Boucle while Maintenant, l'objectif est de travailler avec la boucle while³.

Ce programme modifie le programme précédent afin de s'assurer que l'entrée utilisateur est correcte. Implémentez-le.

```

1 """
2 Définition: Programme interactif de vérification de la présence d'étudiants.
3 réalisé dans le cadre du module R1.03 Bases de la programmation
4 Auteur: Mme Tabary
5 Date: 3/08/2023 Version: 1.0
6 """
7
8 #tableau de la classe
9 listeEtudiants = ["Alice", "Bob", "Charlie", "Diane"]
10 listeEtudiantsAbs = []
11
12 #verification de la présence des étudiants
13 for w in listeEtudiants:
14     print("L'étudiant ",w, "est-il venu en cours ?")
15     reponse = input()
16     #verification de la réponse
17     while reponse != "y" and reponse != "n":

```

³https://docs.python.org/fr/3/reference/compound_stmts.html#the-while-statement


```

18     print("reponse incorrecte, veuillez saisir y ou n")
19     reponse = input()
20     if reponse == "n":
21         listeEtudiantsAbs.append(w)
22
23 #affichage des étudiants présents
24 for w in listeEtudiantsAbs:
25     print("Nom de l'étudiant absent : ",w)
26     for x in listeEtudiantsAbs:
27         print("Nom de l'étudiant absent : ",w)

```

Exercices

- 1♦- Modifier le programme pour demander si le prénom de l'étudiant présent est bien orthographié.
 (optionnel) 2♦ ♦ ♦- Continuer le programme afin de demander une confirmation de l'orthographe avant de valider le prénom. Modifier le prénom dans la liste s'il était mal orthographié.

1.4 Le juste prix

Réaliser ce programme :

Un prix caractérisé par une variable globale est déjà présent dans le programme (variable globale). Le but pour l'utilisateur est de deviner ce prix. Chaque fois que l'utilisateur se trompe, l'ordinateur lui dit si c'est plus ou moins que le prix qu'il a donné. Le joueur a gagné une fois qu'il a trouvé le bon prix. Un compteur d'essai donne le nombre de coups nécessaire pour trouver ce juste prix.

(Optionnel) A votre avis, quelle stratégie de jeu est la meilleure ? et pourquoi sa complexité est de $\log(n)$, avec n la taille de la liste ?

1.4.1 Réaliser ce programme

1.4.2 (Optionnel) Variantes du juste prix

En entrée, prendre une variable de type entier issue de la librairie random

En entrée, prendre une voyelle aléatoire (toujours avec la librairie random)

Avec un nombre de coups limité Le nombre de coups est défini avec une variable globale.

De faire jouer deux joueurs. Chacun choisit une lettre de l'alphabet. Le plus rapide à la trouver a gagné.

De faire jouer x joueurs entre eux. Chacun choisit une lettre de l'alphabet. Le plus rapide à la trouver a gagné.

1.4.3 (Optionnel) Générateur des nombres de la suite de Fibonacci

Sans utiliser de sous-fonction, faite un générateur des nombres de la suite de Fibonacci⁴.

⁴https://fr.wikipedia.org/wiki/Suite_de_Fibonacci

1.5 Bilan (obligatoire)

Lors de ce TP, vous vous êtes arrêté à quel exercice ? _____

Remplir le tableau

Partie savoir faire

- Niveau 1 : Je n'ai pas su l'implémenter. C'est du charabia pour moi.
- Niveau 2 : J'ai lu le sujet. Les couleurs sont jolies. J'ai fini le premier exercice les yeux rivés sur mon clavier pour chercher les touches. Les erreurs de l'interpréteur me semblent incompréhensibles.
- Niveau 3 : J'ai pu avancer à la moitié du sujet, même si c'est difficile et que l'ordi est farceur (comme tous les ordi). Je prends beaucoup de temps à comprendre les erreurs du shell, mais j'y arrive !
- Niveau 4 : J'ai complété le TP avec aisance. La plupart des erreurs du shell me sont compréhensibles.
- Niveau 5 : J'ai complété le TP, exercices optionnels compris ! J'ai une grande agilité⁵ quand je code.

Partie savoir être

- Niveau 1 : Pour finir au plus vite, j'élabore des stratégies (copier directement la réponse ou chercher à camoufler le désintérêt : "Si je n'y arrive pas, c'est que le prof n'est pas venu assez vite me donner la solution").
- Niveau 2 : L'objectif est d'avoir la moyenne sans trop y laisser du temps ou de l'énergie. Je suis bien obligé de faire le TP, même si l'idée de me servir du panel de ressources me semble saugrenue. Si c'est possible de récupérer la réponse (ou de suivre à côté d'un camarade⁶), alors je ne dirai pas non.
- Niveau 3 : Je prends doucement mes marques. Je me suis servi de façon hésitante de plusieurs ressources dispos (shell, camarades, enseignants, manuel, sites web, ou même canard⁷) en cherchant à comprendre leur réponse. J'aimerais un jour pouvoir développer mes propres projets et il faut pour cela que je gagne en autonomie.
- Niveau 4 : Je suis autonome dans l'utilisation des ressources disponibles (shell, camarades, enseignants, manuel, sites web, ou même canard⁷). J'ai même un projet personnel en cours que j'aimerais finir.
- Niveau 5 : J'évolue avec aisance dans cet environnement ! Je fais même partie dorénavant des ressources dispos et j'échange facilement sur ces notions. J'ai plusieurs projets informatiques personnels.

Notions	Niveau atteint (de 1 à 5) Savoir faire	Niveau atteint (de 1 à 5) Savoir être
Instructions simples		
Variables		
Conditionnelles		
Boucle For		
Boucle While		
Respect des règles de bon codage		

SUPER TIP

Pour finir, un site pour s'entraîner/tester les instructions. En français.

Source : <https://www.geeksforgeeks.org/python-tuples/?ref=lbp>

⁵agilité → ne pas utiliser la souris en codant

⁶Mais si, c'est du travail d'équipe : il code et je le soutiens émotionnellement !

⁷https://fr.wikipedia.org/wiki/M%C3%A9thode_du_canard_en_plastique

2 TP2 : Les structures de données

La liste est une collection ordonnée et modifiable. Elle autorise la duplication des items.

Le tuple est une collection ordonnée et immuable. Permet la duplication des items.

Set est une collection non ordonnée, non modifiable et non indexée. Pas d'items en double.

Dictionnaire : collection ordonnée et modifiable. Pas d'items en double.

2.1 Lists

Les éléments de la liste sont ordonnés, modifiables et peuvent être dupliqués.

Pour manipuler une liste, ouvrez un éditeur de texte et écrivez :

```

1  """
2  Définition: Programme qui donne le type des éléments d'une liste.
3  réalisé dans le cadre du module R1.03 Bases de la programmation
4  Auteur: Mme Tabary
5  Date: 3/08/2023    Version: 1.0
6  """
7
8
9  #affectation d'une liste
10 listeTout = ['chat', 1, 'chien', 3, 21,
11             'tortue', 1, True, False,
12             5, 5.5, 'b'
13             ]
14
15 #affichage de la liste
16 print("la liste est :", \
17       listeTout)
18
19 #affectation de la longueur de la liste à la variable longueurListeTout
20 longueurListeTout = len(listeTout)
21
22 #boucle de parcours de la liste
23 for i in range(longueurListeTout):
24     if type(listeTout[i]) == int:
25         print("c'est un entier")
26     elif type(listeTout[i]) == str:
27         print("c'est une chaîne de caractère")
28     elif type(listeTout[i]) == bool:
29         print("c'est un booléen")
30     elif type(listeTout[i]) == float:
31         print("c'est un float")
32     else:
33         print("c'est autre chose")

```

La sortie écran obtenue est la suivante :

```
plproadmin@pdept16:~/Bureau/Cours/sd/ue/r101bp/chp2$ python3 liste.py
la liste est : ['chat', 1, 'chien', 3, 21, 'tortue', 1, True, False, 5, 5.5, 'b']
c'est une chaine de caractère
c'est un entier
c'est une chaine de caractère
c'est un entier
c'est un entier
c'est une chaine de caractère
c'est un entier
c'est un booléen
c'est un booléen
c'est un entier
c'est un float
c'est une chaine de caractère
```

Tip for Code 1 : "Pour un code svelte et en bonne santé"

Votre code commence à prendre de la place.

Par convention, la largeur maximale d'une ligne est de 79 caractères.

Le caractère \ permet de couper des lignes trop longues.

<https://peps.python.org/pep-0008/#maximum-line-length>

Exercices

1◇- Complétez le programme suivant de façon à affecter trois listes : listeChaine, listeBool, listeInt, listeFloat, et listeAutres. Afficher ces listes de façon à générer la sortie terminale suivante.

```
la liste est : ['chat', 1, 'chien', 3, 21, 'tortue', 1, True, False, 5, 5.5, 'b']
c'est une chaine de caractère
c'est un entier
c'est une chaine de caractère
c'est un entier
c'est un entier
c'est une chaine de caractère
c'est un entier
c'est un booléen
c'est un booléen
c'est un entier
c'est un float
c'est une chaine de caractère
les nouvelles listes sont : [1, 3, 21, 1, 5] [True, False] ['chat', 'chien', 'tortue', 'b'] [5.5] []
```

2◇- Modifiez le programme afin de ranger par ordre croissant listeEntier.

Modifiez le programme afin d'inverser listeChaine.

Aide : https://www.w3schools.com/python/python_lists_methods.asp

(optionnel) 3◇ ◇- Modifiez le programme afin d'afficher une liste qui contient l'avant-dernier éléments de chacune des listes (ou rien si la liste est plus petite).

Et afficher une valeur au hasard de cette liste.

Aide : random.choice(maListe) <https://docs.python.org/3/library/random.html>

2.1.1 Strings

Les strings sont des listes un peu particulières⁸.

Ouvrez un éditeur de texte et écrivez :

```

1  """
2  Définition: Programme qui transforme une liste en une chaîne de caractère.
3  réalisé dans le cadre du module R1.03 Bases de la programmation
4  Auteur: Mme Tabary
5  Date: 3/08/2023    Version: 1.0
6  """
7
8
9
10 #affectation d'une liste
11 listeTest = ['J\'étudie dans la formation du','BUT', 'Sciences', 'des', 'Données' ]
12
13 #création d'une chaîne de caractère à partir d'une liste
14 monString = " ".join(listeTest)
15
16 print(monString)
17
18 #découpage de la chaîne de caractère en liste
19 listeFinale = monString.split()
20
21 #affichage de la liste
22 print("la liste est :", listeFinale)

```

La sortie écran que vous devez obtenir est la suivante :

```

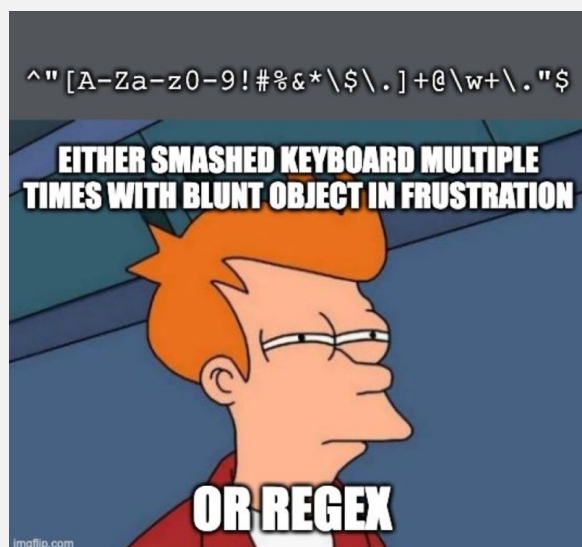
J'étudie dans la formation du BUT Sciences des Données
la liste est : ["J'étudie", 'dans', 'la', 'formation', 'du', 'BUT', 'Sciences', 'des', 'Données']

```

Tip for Code 2 : "Les expressions régulières"

Réservée aux plus aguerris, une expression régulière ou regex est une chaîne de caractères qui décrit, selon une syntaxe précise, un ensemble de chaînes de caractères possibles.

Il existe sur internet des petits jeux pour apprendre à les maîtriser comme <https://regexcrossword.com/>.



⁸https://www.w3schools.com/python/python_strings_methods.asp

Exercices

1◇- Complétez le programme suivant de façon à partir de la liste finale :

- a) Affichez le nombre de mots de la liste
- b) Affichez le nombre d'occurrences du mot "BUT"
- c) Rajouter entre "J'étudie" et "dans" les éléments "à", "Dole"
- d) Supprimer les deux derniers éléments de la liste
- e) Rajouter des "euh" à chaque espace

Puis, recréez une phrase (string) à partir de cette liste et affichez là. Le résultat devrait être ainsi :

```
J'étudie dans la formation du BUT Sciences des Données
la liste est : ["J'étudie", 'dans', 'la', 'formation', 'du', 'BUT', 'Sciences', 'des', 'Données']
le nombre de mots de la liste est : 9
le nombre d'occurrences du mot "BUT" est : 1
la liste est : ["J'étudie", 'à', 'Dole', 'dans', 'la', 'formation', 'du', 'BUT', 'Sciences', 'des', 'Données']
le string devient : J'étudie euh à euh Dole euh dans euh la euh formation euh du euh BUT euh Sciences
```

2◇- Complétez le programme suivant de façon à partir d'un string obtenu

- a) mettez le string en majuscule (aide : upper()^a)
- b) remplacez les "euh" par un espace " " (aide : replace())
- c) afficher si le string contient le mot "DOLE" (aide : rfind())

Puis, recréez une liste à partir de ce string et affichez là en minuscule.

(Optionnel) 3◇ ◇ ◇- Modifier la comparaison avec "DOLE" de façon à ce qu'elle soit insensible à la case.

^a<https://www.geeksforgeeks.org/python-string-upper/>

2.2 Tuples

Les éléments d'un tuple sont ordonnés, immuables et peuvent être dupliqués. En mathématiques, on parle de p-uplets, avec p ne nombre d'éléments <https://python-reference.readthedocs.io/en/latest/docs/tuple/>.

Ouvrez un éditeur de texte et écrivez :

```
1
2 """
3 Définition: Programme de manipulation de tuples.
4 réalisé dans le cadre du module R1.03 Bases de la programmation
5 Auteur: Mme Tabary
6 Date: 3/08/2023    Version: 1.0
7 """
8
9 #affectation d'un tuple
10 dépensesTuple = (5, 21, 101, 423)
11
12 #affectation de variables
```

```

13 (gaz, forfait, course, loyer) = depensesTuple
14
15 #affichage des variables
16 print ("le tuple est :")
17 print(course)
18 print(forfait)
19 print(loyer)
20
21 #affectation tuple sur 3 mois
22 mesPrevisionsTroisMois = depensesTuple * 3
23
24 #affichage du nouveau tuple
25 print(mesPrevisionsTroisMois)

```

La sortie écran que vous devez obtenir est la suivante :

```

le tuple est :
5
21
423
(5, 21, 100, 423, 5, 21, 100, 423, 5, 21, 100, 423)

```

Exercices

1♦- Complétez le programme précédent de façon à fournir :

- a) la somme des dépenses
- b) la dépense la plus importante
- c) la dépense la moins importante
- d) le nombre de loyers payés (montant 423)
- e) le nombre de dépenses
- f) la moyenne des dépenses
- g) la somme des dépenses

L'affichage terminal obtenu devrait être le suivant :

```

le tuple est :
101
21
423
(5, 21, 101, 423, 5, 21, 101, 423, 5, 21, 101, 423)
la somme des dépenses est de : 1650
la dépense la plus importante est de : 423
la dépense la moins importante est de : 5
le nombre de dépenses est de : 12
la moyenne des dépenses est de : 137.5
la somme des dépenses est de : 1650

```

Aide : <https://python-reference.readthedocs.io/en/latest/docs/tuple/>

Tip for Code 3 : psychologie d'un tuple

Les tuples étant non modifiables, que se passe-t-il alors avec un tuple contenant des objets modifiables comme des listes ?

Analyse du code suivant :

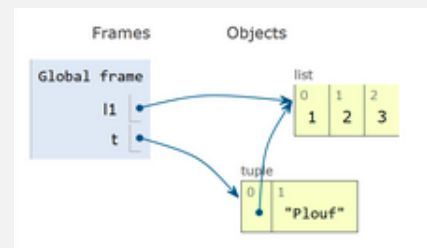
```

1
2
3 "Exemple de tuple avec une liste à l'intérieur
4 liste1 = [0, 1, 2]
5 tuple1 = (liste1, "Plouf")
6
7 print ("mon tuple =", tuple1)
8 #sortie écran : "mon tuple = ([0, 1, 2], 'Plouf')"
9
10
11 #Objectif : avoir "mon tuple = ([1, 1, 2], 'Plouf')"
12 # Mais on ne peut pas modifier un tuple :
13 # tuple1[0] = [1, 1, 2] : ERREUR
14
15 #On peut modifier une liste :
16 liste1[0] = 1
17
18 print ("apres modification de l1, on a mon Tuple=",
19       tuple1)
20 #sortie écran : "apres modification de l1, on a mon
21 Tuple= ([1, 1, 2], 'Plouf')"
22
23 #Observation : on peut modifier la liste à l'
24 intérieur du tuple

```

Il existe deux grands types de copie :

- La copie par valeur recrée un objet identique. Deux objets d'adresse différente mais de valeur identique sont créés.
- La copie par référence copie l'adresse (ou référence) de l'objet. Il n'existe donc qu'un seul objet.



La liste l1 pointe vers le même objet que l'élément du tuple d'indice 0. C'est une copie par référence (même adresse). Donc, Dans notre cas, le premier élément du tuple pointe vers une liste modifiable.

2.3 Sets

Les éléments de l'ensemble sont non ordonnés, non modifiables et n'autorisent pas les valeurs en double.

Pour manipuler un ensemble, ouvrez un éditeur de texte et écrivez :

```

1
2 """
3 Définition: Programme de manipulation des ensembles.
4 réalisé dans le cadre du module R1.03 Bases de la programmation
5 Auteur: Mme Tabary
6 Date: 3/08/2023 Version: 1.0
7 """
8
9
10 #affectation d'une chaine de caractère
11 adn = "atctcgatcgatcgctagctagctcgccatacgtacgactacgt"
12 #transformation de la chaine de caractère en ensemble
13 adnSet = set(adn)
14 #affichage de l'ensemble
15 print(set(adnSet))
16
17 #affectation d'un ensemble
18 aaEpinards = {"gl", "tyr", "a", True, 3.14}
19 #affichage de l'ensemble

```



```
20 print(aaEpinards)
```

La sortie écran obtenue est la suivante :

```
plproadmin@pdept16:~/Bureau/Cours/sd/ue/r101bp/chp2$ python3 ens3.py
{'c', 't', 'a', 'g'}
{'gl', 'a', 'tyr'}
('abc', 34, True, 40, 'UFR', 3.14)
```

Tip for Code 4 : "Les duplications dans les ensembles"

Les ensembles ne peuvent pas comporter deux éléments ayant la même valeur. Les valeurs en double seront ignorées. Le programme suivant :

```
1 unEnsemble = {"chat", "chien", "mésange", "chat"}
2
3 print(unEnsemble)
```

donnera la sortie écran suivante :

```
plproadmin@pdept16:~/Bureau/Cours/sd/ue/r101bp/chp2$ python3 ens.py
{'mésange', 'chien', 'chat'}
```

A votre avis, que donnera le programme suivant ?

```
1 unEnsemble = {"chat", "chien", "mésange", "chat", 1, True}
2
3 print(unEnsemble)
```

Exercice

1◇- A l'aide du programme précédent et de votre ordinateur, complétez le tableau suivant

set

Instructions	Sortie écran
<code>s = {3, 4, "Plouf", (1, 3)}</code> <code>print(s)</code>	
<code>s2 = {3.14, [1, 2]}</code> <code>print(s)</code>	
<code>print((set((2, 2, 2, 1))))</code>	
<code>s3 = set("BUTSDS1")</code> <code>print(s3)</code>	
<code>s3.add(1)</code> <code>print(s3)</code>	
<code>s3.discard('1')</code> <code>print(s3)</code>	
<code>s3.discard('t')</code> <code>print(s3)</code>	
<code>print(s3.intersection(s))</code>	

Tip for Code 5 : Etre ou ne pas être (hashable)

Commençons avec un rappel de vocabulaire :

Un objet mutable est ainsi un objet qui peut être modifié, dont on peut changer les propriétés une fois qu'il a été défini.^a

A l'inverse, **un objet hashable** ne peut être modifié une fois qu'il a été défini.^b

Les sets ne peuvent contenir que des objets hachables. Avoir des objets hashables optimise l'accès à chaque élément du set. Les objets hachables sont les chaînes de caractères, les tuples, les entiers, les floats, les booléens et les frozensets. Les objets non hachables que l'on connaît sont les listes, les sets et les dictionnaires.

^a<https://docs.python.org/3/glossary.html#term-mutable>

^b<https://docs.python.org/3/glossary.html#term-hashable>

2.4 Dictionnaires

Les dictionnaires permettent de manipuler des structures complexes. **Les dictionnaires sont des collections non ordonnées d'objets**. Il s'agit d'objets correspondance (mapping objects en anglais) ou tableaux associatifs. En effet, dans un même dictionnaire chaque valeur d'objet est accessible par une clé.

Pour manipuler un ensemble, ouvrez un éditeur de texte et écrivez :

```
1 #affectation d'un dictionnaire
2 thisdict = {
3     "nom": "Tabary",
4     "profession": {"Enseignant", "chercheur", "Directice d'études", "Maitre de conférences"},
5     "age": 1988
6 }
7
8 #affichage du dictionnaire
9 print("le disctionnaire est ",thisdict)
10
11 #affichage du dictonnaire clé par clé
12 for key in thisdict:
13     print(key, "(cle) : ", thisdict[key], "(valeur)")
14
15 #vérification de l'existence d'une clé
16 if "age" in thisdict:
17     print("La clé 'age' existe pour thisdict")
18
19 #affichage du dictionnaire ordonné
20 print(sorted(thisdict))
```

La sortie écran obtenue est la suivante :

```
le disctionnaire est {'nom': 'Tabary', 'profession': {'chercheur', 'Maitre de c
onférences', 'Enseignant', "Directice d'études"}, 'age': 1988}
nom (cle) : Tabary (valeur)
profession (cle) : {'chercheur', 'Maitre de conférences', 'Enseignant', "Direct
ice d'études"} (valeur)
age (cle) : 1988 (valeur)
La clé 'age' existe pour thisdict
['age', 'nom', 'profession']
```

Exercice

1◇- Modifier le code précédent afin d'enlever la valeur enseignant chercheur au dictionnaire.

2◇ ◇- Rajouter le dictionnaire suivant au code

```
23  thisdict2 = {
24      "nom": "Mohammed",
25      "profession": {"Développeur web", "Enseignant", "Manager"},
26      "age": 1987
27  }
28  mesProfs = [thisdict, thisdict2]
```

- affichez les noms de vos professeurs grâce à une boucle
- affichez l'ensemble des professions communes aux deux professeurs

2.5 Projets (optionnels)

La recherche dichotomique prend en entrée une liste triée T et un élément v . Elle donne en résultat l'index, tel que $T[index] = v$.

2.5.1 Recherche simple

On procède comme suit :

- On compare v à chaque élément de la liste
- S'il est égal à v , on a fini
- Sinon, on prend l'élément suivant
- S'il est supérieur, on affiche l'information "pas d'éléments" sur le terminal

Donnez une preuve de terminaison, et la complexité de ce programme.

Programmez un algorithme de recherche dichotomique.

2.5.2 Recherche dichotomique

On procède comme suit :

- On compare v à l'élément du milieu de la liste.
- S'il est égal à v , on a fini.
- Sinon, s'il est inférieur, il faut chercher dans la première moitié de la liste. On retourne à l'étape 1 avec la liste réduite.
- S'il est supérieur, on fait de même avec la seconde moitié de la liste.

Programmez un algorithme de recherche dichotomique.

Donnez une preuve de terminaison, et la complexité de ce programme.

2.6 Bilan (obligatoire)

Lors de ce TP, vous vous êtes arrêté à quel exercice ? _____

Remplir le tableau

Partie savoir faire

- Niveau 1 : Je n'ai pas su l'implémenter. C'est du charabia pour moi.
- Niveau 2 : J'ai lu le sujet. Les couleurs sont jolies. J'ai fini le premier exercice les yeux rivés sur mon clavier pour chercher les touches. Les erreurs de l'interpréteur me semblent incompréhensibles.
- Niveau 3 : J'ai pu avancer à la moitié du sujet, même si c'est difficile et que l'ordi est farceur (comme tous les ordi). Je prends beaucoup de temps à comprendre les erreurs du shell, mais j'y arrive !
- Niveau 4 : J'ai complété le TP avec aisance. La plupart des erreurs du shell me sont compréhensibles.
- Niveau 5 : J'ai complété le TP, exercices optionnels compris ! J'ai une grande agilité⁹ quand je code.

Partie savoir être

- Niveau 1 : Pour finir au plus vite, j'élabore des stratégies (copier directement la réponse ou chercher à camoufler le désintérêt : "Si je n'y arrive pas, c'est que le prof n'est pas venu assez vite me donner la solution").
- Niveau 2 : L'objectif est d'avoir la moyenne sans trop y laisser du temps ou de l'énergie. Je suis bien obligé de faire le TP, même si l'idée de me servir du panel de ressources me semble saugrenue. Si c'est possible de récupérer la réponse (ou de suivre à côté d'un camarade¹⁰), alors je ne dirai pas non.
- Niveau 3 : Je prends doucement mes marques. Je me suis servi de façon hésitante de plusieurs ressources dispos (shell, camarades, enseignants, manuel, sites web, ou même canard¹¹) en cherchant à comprendre leur réponse. J'aimerais un jour pouvoir développer mes propres projets et il faut pour cela que je gagne en autonomie.
- Niveau 4 : Je suis autonome dans l'utilisation des ressources disponibles (shell, camarades, enseignants, manuel, sites web, ou même canard⁷). J'ai même un projet personnel en cours que j'aimerais finir.
- Niveau 5 : J'évolue avec aisance dans cet environnement ! Je fais même partie dorénavant des ressources dispos et j'échange facilement sur ces notions. J'ai plusieurs projets informatiques personnels.

Notions	Niveau atteint (de 1 à 5) Savoir faire	Niveau atteint (de 1 à 5) Savoir être
Liste		
Tuple		
Set		
Dictionnaire		
Respect des règles de bon codage		

⁹agilité → ne pas utiliser la souris en codant

¹⁰Mais si, c'est du travail d'équipe : il code et je le soutiens émotionnellement !

¹¹https://fr.wikipedia.org/wiki/M%C3%A9thode_du_canard_en_plastique

3 TP3 : Lecture / écriture de fichier

Un DataFrame est une structure de données étiquetée à deux dimensions avec des colonnes de types potentiellement différents. Un DataFrame peut-être une feuille de calcul, une table SQL ou un dictionnaire d'objets de série.

Ce TP vous invite à manipuler des dataframe à partir de fichiers externes.

3.1 Fichiers de type texte (.txt)

Pour manipuler un fichier texte, copiez/collez le fichier texte "biblo1.txt" dans le même fichier que le programme et écrivez ce programme dans un éditeur de texte :

```
1 """
2 Définition: Programme qui lit un texte et affiche les lignes qui contiennent un point.
3 réalisé dans le cadre du module R1.03 Bases de la programmation
4 Auteur: Mme Tabary
5 Date: 7/08/2023    Version: 1.0
6 """
7
8 # ouverture du fichier en lecture
9 fichier = open("bilbo1.txt", "r")
10 print(fichier)
11
12 # lecture du fichier ligne par ligne
13 i = 0
14 for ligne in fichier:
15     print("Ligne ", i, " :", ligne)
16     i = i + 1
17     # recherche du caractère point dans la ligne
18     if ( "." in ligne):
19         print("Il y a un point dans la ligne ", i, " a la place ", ligne.find("."))
20 # fermeture du fichier
21 fichier.close()
```

La sortie écran obtenue est la suivante :

```
plproadmin@pdept16:~/Bureau/Cours/sd/ue/r101bp/chp3$ python3 texte.py
<_io.TextIOWrapper name='bilbo1.txt' mode='r' encoding='UTF-8'>
Ligne 0 : Dans un trou vivait un hobbit.
Il y a un point dans la ligne 1 a la place 29
```

On retrouve les méthodes de manipulation des strings (chaînes de caractère), vues dans le chapitre précédent, et détaillées dans le manuel ¹².

¹²https://www.w3schools.com/python/python_ref_string.asp

Tip for Code 1 : "les Droits"

Les droits sur le fichier manipuler sont déclinés par les lettres suivantes :

r, pour une ouverture en lecture (READ).

w, pour une ouverture en écriture (WRITE), à chaque ouverture le contenu du fichier est écrasé. Si le fichier n'existe pas python le crée.

a, pour une ouverture en mode ajout à la fin du fichier (APPEND). Si le fichier n'existe pas python le crée.

b, pour une ouverture en mode binaire.

t, pour une ouverture en mode texte.

x, crée un nouveau fichier et l'ouvre pour écriture

WARNING

Tout fichier ouvert (avec open) doit être fermé (avec close).

Tout oubli à ce sujet, vaudra une division par 2 de la note de l'exercice en question.

Exercices

1◇- Le premier exercice vous invite à manipuler des fichiers textes.

a) Mettre les fichiers textes "bilbo1.txt", "bilbo2.txt" et "bilbo3.txt" dans un dossier intitulé "fichiersTexte" dans le même dossier que votre programme.

b) Créer un nouveau fichier texte vide, appelé "bilbo.txt".

c) Concaténer les 3 fichiers en un nouveau fichier "blibo.txt", lui-même placé dans le dossier "fichiersTexte".

2◇- Modifier votre code de façon à créer le fichier texte "bilbo.txt" s'il n'est pas existant.

3◇ ◇- Modifier votre code afin de changer le mot "hobbit" par "Periannath" (langue elfique).

(Optionnel)4◇ ◇- Modifier votre programme afin d'intégrer, si ce n'est pas déjà fait, la librairie de gestion des fichiers "import os"^a pour importer l'adresse du fichier courant.

(Optionnel)5◇ ◇ ◇- Créer un programme qui prend en entrée un dossier. Et en sortie, ce programme affiche la hiérarchie des éléments au sein de ce dossier, avec le type de ces éléments (fichier, dossier, exécutable, ou autres).

^a<https://docs.python.org/fr/3/library/os.html>

3.2 Les fichiers csv

Un fichier CSV (Comma-separated values) est un type de fichier associé à Excel. Plutôt que de stocker les informations en colonnes, les fichiers CSV les stockent en les séparant par des points-virgules.

Le csv utilisé lors de ce TP est disponible sur le site de l'INSEE¹³ sous le lien : Fichier allégé par département depuis 2000.

¹³<https://www.insee.fr/fr/statistiques/2540004?sommaire=4767262#consulter>

Fichier allégé par département depuis 2000

(csv, 3 Mo)



Télécharger l'archive dans un dossier intitulé "fichiersCSV", dans le même répertoire que celui contenant votre dossier "fichiersTexte".

Ensuite, copier le programme suivant :

```

1  """
2  Définition: Programme qui compte le nombre de lignes dans un fichier csv
3              et qui énumère les prénoms féminins des personnes nées en 2021,
4              dont le prénom a été donné 4 fois
5  réalisé dans le cadre du module R1.03 Bases de la programmation
6  Auteur: Mme Tabary
7  Date: 7/08/2023    Version: 1.0
8  """
9
10 # import csv pour lire et écrire dans un fichier csv
11 import csv
12
13 # ouverture du fichier
14 with open('fichiersCSV/Dpt39depuis2000.csv') as moncsv:
15     # lecture du fichier csv
16     csv_reader = csv.reader(moncsv, delimiter=',')
17     # compteur de lignes
18     line_count = 0
19     for ligne in csv_reader:
20         if line_count == 0:
21             # affichage du nom des colonnes
22             print(f'Les colonnes sont : {", ".join(ligne)}')
23             # incrémentation du compteur de lignes
24             line_count += 1
25
26             # affichage des prénoms féminins des personnes nées en 2021, dont le prénom a été donné 4 fois
27             if ligne[0].split(";")[2] == "2021" and ligne[0].split(";")[0] == "2" and ligne[0].split(";")[4]
28             == "4":
29                 print("Prénom : ", ligne[0].split(";")[1])
30             # affichage du nombre de lignes total
31             print(f'Au total, il y a {line_count} lignes.')

```

La sortie écran obtenue est la suivante :

```
plproadmin@pdept16:~/Bureau/Cours/sd/ue/r101bp/chp3$ python3 EcritureLecturecsv.py
Les colonnes sont : sexe;preusuel;annais;dpt;nombre
Prénom : ALIX
Prénom : ANAÏS
Prénom : ANNA
Prénom : CAPUCINE
Prénom : CHARLIE
Prénom : ÉLISE
Prénom : HORTENSE
Prénom : IRIS
Prénom : JULIA
Prénom : JULIETTE
Prénom : LILY
Prénom : LIVIA
Prénom : LOUANE
Prénom : MANON
Prénom : MILA
Prénom : MYA
Prénom : SOLINE
Prénom : VICTOIRE
Au total, il y a 4787 lignes.
```

Exercices

1◇- Le premier exercice vous invite à manipuler le programme précédent de façon à :

- Enumérer les prénoms féminins de 2021 qui apparaissent -cette fois- moins de 5 fois
- Ecrire le résultat dans un fichier .txt, avec la mention "prénom rare" à côté, si le prénom apparaît moins de 4 fois.




(Optionnel) 2◇- Concaténer les deux cvs du Jura (39) et du Doubs (25) en un seul csv

(Optionnel) 3◇- Pourquoi vous ne devez pas appeler votre fichier python csv.py ?

3.3 Fichiers de type xlsc

Le fichier xlsx est l'extension par défaut des fichiers créés avec la version 2007 du logiciel Excel.

Le xlsx utilisé lors de ce TP est disponible sur le site de l'INSEE¹⁴ sous le lien : France hors Mayotte.

France hors Mayotte		
(pdf, 12 Mo)		
(xlsx, 2 Mo)		
(csv, 971 Ko)		

Télécharger le fichier dans un dossier intitulé "fichiersXLSX", dans le même répertoire que celui contenant vos dossiers "fichiersTexte" et "fichiersCSV".

Ensuite, copier le programme suivant :

```
1 """
2 Définition: Programme qui affiche la population des communes dont le nom commence par Dol
3 réalisé dans le cadre du module R1.03 Bases de la programmation
4 Auteur: Mme Tabary
5 Date: 7/08/2023    Version: 1.0
6 """
7
```

¹⁴<https://www.insee.fr/fr/statistiques/6011070?sommaire=6011075>


```

8 # import de la bibliothèque pandas
9 import pandas as panda
10
11 # affectation des noms de colonnes qui nous intéressent
12 colonnes = ['nomCommune', 'population']
13
14 # lecture du fichier excel
15 df = panda.read_excel(open('fichiersXLSX/ensemble.xlsx', 'rb'),
16                        sheet_name='Communes',
17                        usecols=[6,9],
18                        names = colonnes)
19
20 # affichage de la population des communes dont le nom commence par Dol
21 for idx, x in enumerate(df.nomCommune):
22     if type(x) == str and x.startswith('Dol'):
23         print("Population de", x, "égale ", df.population[idx])

```

Exercice

Pourquoi il est déconseillé d'écrire L 22 : " if x.startswith('Dol') and type(x) == str:" ?

La sortie écran obtenue est la suivante :

```

plproadmin@pdept16:~/Bureau/Cours/sd/ue/r101bp/chp3$ python3 EcritureLecturexlsx.py
Population de Dolignon égale 48
Population de Dolancourt égale 137
Population de Dolus-d'Oléron égale 3206
Population de Dol-de-Bretagne égale 6070
Population de Dolus-le-Sec égale 696
Population de Dolomieu égale 3263
Population de Dole égale 24604
Population de Dolmayrac égale 731
Population de Dolcourt égale 135
Population de Dolving égale 352
Population de Dolleren égale 490
Population de Dollon égale 1483
Population de Dolaincourt égale 100
Population de Dollot égale 319

```

On retrouve les méthodes de manipulation des strings (chaînes de caractère), vues dans le chapitre précédent, et détaillées dans le manuel ¹⁵.

De plus, on utilise la méthode `enumerate` ¹⁶ pour accéder aux index de la boucle (voir chapitre précédent).

¹⁵https://www.w3schools.com/python/python_ref_string.asp

¹⁶<https://docs.python.org/3/library/functions.html#enumerate>

Tip for Code 2 : "Panel Data"

La bibliothèque logicielle open-source Pandas est spécifiquement conçue pour la manipulation et l'analyse de données en langage Python. Elle est à la fois performante, flexible et simple d'utilisation.

Grâce à Pandas, le langage Python permet enfin de charger, d'aligner, de manipuler ou encore de fusionner des données. Les performances sont particulièrement efficaces quand le code source back-end est écrit en C ou en Python.

Le manuel est disponible à cette adresse : <https://pandas.pydata.org/docs/reference/api/pandas.ExcelFile.html>

Exercices

1◇- Le premier exercice vous invite à manipuler le programme précédent de façon à : Afficher sur le terminal la population totale des villes commençant par "Dol"

(Optionnel)2◇ ◇- On s'intéresse en suite aux communes dont la commune pôle est Mièges (Mièges comprise), dans la feuille "Communes associées ou déléguées". Créer un fichier texte contenant :

- a) le nombre de ces communes
- b) leur nom, leur population municipale et leur population totale
- c) si leur population comptée à part est null, rajouter la mention "urbain"
- d) afficher la population totale

(Optionnel)3◇ ◇ ◇- Créer un nouveau fichier xlsx, de façon à avoir les colonnes de la feuille "Communes associées ou déléguées", avec le numéro de Canton (dans la feuille "Communes").

Vous rajouterez également une visualisation de la population de votre choix en utilisant la bibliothèque plot ^a.

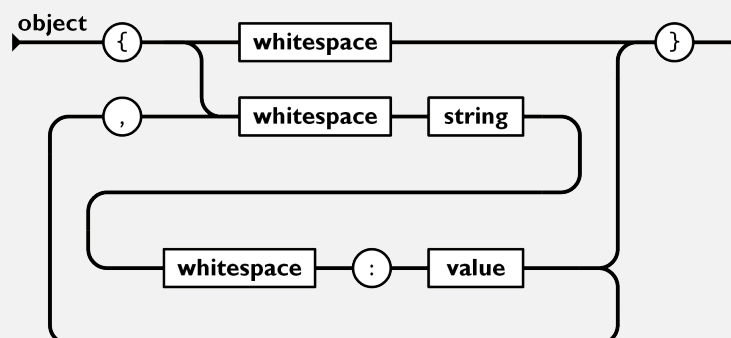
(Optionnel)4◇- Pourquoi vous ne devez pas appeler votre fichier python csv.py ?

^ahttps://pandas.pydata.org/docs/user_guide/visualization.html

3.4 Fichiers de type json

JSON

Le JavaScript Object Notation (JSON) est un format standard utilisé pour représenter des données structurées de façon semblable aux objets Javascript. Un json est représenté par une liste de dictionnaires.



Pour cet exercice, copier le programme suivant :

```

1 """
2 Définition: Programme qui crée un fichier json contenant les cours
3 réalisé dans le cadre du module R1.03 Bases de la programmation
4 Auteur: Mme Tabary
5 Date: 7/08/2023    Version: 1.0
6 """
7
8 # import pandas pour lire et écrire dans un fichier json
9 import pandas as panda
10
11 # definition d'un dictionnaire
12 mesCours = [{'Cours': 'Bases de la programmation', 'heures': 20, 'Durée': '30jours'},
13             {'Cours': 'Bases de Données relationnelles', 'heures': 30, 'Durée': '40jours'}]
14
15 # création d'un dataframe à partir du dictionnaire
16 df = panda.DataFrame(mesCours)
17 # création d'un fichier json à partir du dataframe
18 df.to_json('mesCoursFormatJSON.json')
19 # affichage du dataframe sur le terminal
20 print(df)

```

La sortie écran obtenue est la suivante :

```

plproadmin@pdept16:~/Bureau/Cours/sd/ue/r101bp/chp3$ python3 EcritureLecturejson.py
      Cours  heures  Durée
0  Bases de la programmation    20  30jours
1  Bases de Données relationnelles    30  40jours

```

Exercice

1◇- A partir du manuel^a et du programme précédent, rajouter le numéro de semestre correspondant au cours et les notes de ces matières.

^a<https://pythonbasics.org/pandas-json/>

3.5 Bilan (obligatoire)

Lors de ce TP, vous vous êtes arrêté à quel exercice ? _____

Remplir le tableau

Partie savoir faire

- Niveau 1 : Je n'ai pas su l'implémenter. C'est du charabia pour moi.
- Niveau 2 : J'ai lu le sujet. Les couleurs sont jolies. J'ai fini le premier exercice les yeux rivés sur mon clavier pour chercher les touches. Les erreurs de l'interpréteur me semblent incompréhensibles.
- Niveau 3 : J'ai pu avancer à la moitié du sujet, même si c'est difficile et que l'ordi est farceur (comme tous les ordi). Je prends beaucoup de temps à comprendre les erreurs du shell, mais j'y arrive !
- Niveau 4 : J'ai complété le TP avec aisance. La plupart des erreurs du shell me sont compréhensibles.
- Niveau 5 : J'ai complété le TP, exercices optionnels compris ! J'ai une grande agilité¹⁷ quand je code.

Partie savoir être

- Niveau 1 : Pour finir au plus vite, j'élabore des stratégies (copier directement la réponse ou chercher à camoufler le désintérêt : "Si je n'y arrive pas, c'est que le prof n'est pas venu assez vite me donner la solution").
- Niveau 2 : L'objectif est d'avoir la moyenne sans trop y laisser du temps ou de l'énergie. Je suis bien obligé de faire le TP, même si l'idée de me servir du panel de ressources me semble saugrenue. Si c'est possible de récupérer la réponse (ou de suivre à côté d'un camarade¹⁸), alors je ne dirai pas non.
- Niveau 3 : Je prends doucement mes marques. Je me suis servi de façon hésitante de plusieurs ressources dispos (shell, camarades, enseignants, manuel, sites web, ou même canard¹⁹) en cherchant à comprendre leur réponse. J'aimerais un jour pouvoir développer mes propres projets et il faut pour cela que je gagne en autonomie.
- Niveau 4 : Je suis autonome dans l'utilisation des ressources disponibles (shell, camarades, enseignants, manuel, sites web, ou même canard⁷). J'ai même un projet personnel en cours que j'aimerais finir.
- Niveau 5 : J'évolue avec aisance dans cet environnement ! Je fais même partie dorénavant des ressources dispos et j'échange facilement sur ces notions. J'ai plusieurs projets informatiques personnels.

Notions	Niveau atteint (de 1 à 5) Savoir faire	Niveau atteint (de 1 à 5) Savoir être
txt		
csv		
xlsx		
json		
Respect des règles de bon codage		

¹⁷agilité → ne pas utiliser la souris en codant

¹⁸Mais si, c'est du travail d'équipe : il code et je le soutiens émotionnellement !

¹⁹https://fr.wikipedia.org/wiki/M%C3%A9thode_du_canard_en_plastique

4 TP4 : Entraînement contrôle

Ces exercices sont issus de <https://adventofcode.com/2015/day/1>.

4.1 Exercice 1. Année 2015, jour 1

Vous avez en entrée un fichier texte `exo1.txt` contenant des parenthèses. En sortie, vous devrez afficher la valeur d'une variable sur votre terminal de commande.

4.1.1 première partie

— Jour 1 : Pas tout à fait Lisp —

Le Père Noël essaie de livrer des cadeaux dans un grand immeuble, mais il n'arrive pas à trouver le bon étage - les instructions qu'il a reçues sont un peu confuses. Il commence par le rez-de-chaussée (étage 0) et suit les instructions un caractère à la fois.

Une parenthèse ouvrante, `(`, signifie qu'il doit monter d'un étage, et une parenthèse fermante, `)`, signifie qu'il doit descendre d'un étage.

L'immeuble est très haut et le sous-sol est très profond ; il ne trouvera jamais l'étage supérieur ni l'étage inférieur.

Par exemple, `((`) et `()()` :

`((`) et `()()` donnent tous deux l'étage 0.

`((((` et `((()((` donnent tous deux l'étage 3.

`(((((` donnent également l'étage 3.

`()` et `))()` donnent tous deux l'étage -1 (le premier niveau du sous-sol).

`)))` et `))())` donnent tous deux l'étage -3.

A quel étage les instructions conduisent-elles le Père Noël ?

4.1.2 seconde partie

Maintenant, avec les mêmes instructions, trouvez la position du premier personnage qui le fait entrer dans le sous-sol (étage -1). Le premier caractère dans les instructions a la position 1, le deuxième caractère a la position 2, et ainsi de suite.

Par exemple :

`)` le fait entrer dans le sous-sol à la position du caractère 1.

`()()` le fait entrer dans le sous-sol à la position 5.

Quelle est la position du caractère qui fait entrer le Père Noël en premier dans la cave ?

4.2 Exercice 2. Année 2015, jour 2

Vous avez en entrée un fichier texte `exo2.txt` contenant les données $a \times b \times c$ avec a, b et c des nombres. En sortie, vous devrez afficher la valeur d'une variable sur votre terminal de commande.

4.2.1 première partie

— Jour 2 : On m'a dit qu'il n'y aurait pas de maths...

Les lutins sont à court de papier d'emballage et doivent donc passer une commande pour en obtenir d'autres. Ils ont une liste des dimensions (longueur l , largeur w , et hauteur h) de chaque cadeau, et ne veulent commander que la quantité exacte dont ils ont besoin.

Heureusement, chaque cadeau est une boîte (un prisme rectangulaire droit parfait), ce qui facilite le calcul du papier d'emballage nécessaire pour chaque cadeau : trouver la surface de la boîte, qui est $2*l*w + 2*w*h + 2*h*l$. Les lutins ont également besoin d'un peu de papier supplémentaire pour chaque cadeau : la surface du plus petit côté.

C'est l'aire du plus petit côté :

Un cadeau de dimensions $2 \times 3 \times 4$ nécessite $2*6 + 2*12 + 2*8 = 52$ mètres carrés de papier d'emballage plus 6 mètres carrés de mou, soit un total de 58 mètres carrés.

Un cadeau de dimensions $1 \times 1 \times 10$ nécessite $2*1 + 2*10 + 2*10 = 42$ mètres carrés de papier d'emballage plus 1 mètre carré de jeu, pour un total de 43 mètres carrés.

Tous les nombres de la liste des lutins sont exprimés en pieds.

Combien de mètres carrés de papier d'emballage doivent-ils commander ?

4.2.2 seconde partie

Les lutins sont également à court de ruban. Les rubans étant tous de la même largeur, ils n'ont qu'à se préoccuper de la longueur qu'ils doivent commander, qu'ils voudraient encore une fois exacte.

Le ruban nécessaire pour emballer un cadeau correspond à la distance la plus courte autour de ses côtés, ou au plus petit périmètre d'une face. Chaque cadeau a également besoin d'un nœud fait de ruban ; le nombre de pieds de ruban requis pour un nœud parfait est égal au volume en pieds cubes du cadeau. Ne demandez pas comment ils font le nœud, ils ne vous le diront jamais.

Voici un exemple :

Un cadeau de dimensions $2 \times 3 \times 4$ nécessite $2+2+3+3 = 10$ pieds de ruban pour envelopper le cadeau, plus $2*3*4 = 24$ pieds de ruban pour le nœud, soit un total de 34 pieds.

Un cadeau aux dimensions $1 \times 1 \times 10$ nécessite $1+1+1+1 = 4$ pieds de ruban pour envelopper le cadeau plus $1*1*10 = 10$ pieds de ruban pour le nœud, soit un total de 14 pieds.

Combien de mètres de ruban doivent-ils commander au total ?

4.3 Exercice 3. Année 2015, jour 3

Vous avez en entrée un fichier texte `exo3.txt`. En sortie, vous devrez afficher la valeur d'une variable sur votre terminal de commande.

4.3.1 première partie

— Jour 3 : Des maisons parfaitement sphériques dans le vide —

Le Père Noël livre des cadeaux à une grille bidimensionnelle infinie de maisons.

Il commence par livrer un cadeau à la maison située à son point de départ, puis un elfe du pôle Nord l'appelle par radio pour lui dire où se déplacer ensuite. Les déplacements se font toujours exactement d'une maison vers le nord (\uparrow), le sud (\downarrow), l'est (\rightarrow) ou l'ouest (\leftarrow). Après chaque déplacement, il livre un autre cadeau à la maison de son nouvel emplacement.

Cependant, le lutin qui se trouve au pôle Nord a bu un peu trop de lait de poule, et ses indications sont donc un peu erronées, de sorte que le Père Noël finit par visiter certaines maisons plus d'une fois. **Combien de maisons reçoivent au moins un cadeau ?**

Par exemple :

> livre des cadeaux à 2 maisons : l'une au point de départ, l'autre à l'est.

^ >v < livre des cadeaux à 4 maisons dans un carré, dont deux fois à la maison située à son emplacement de départ/arrivée.

^v ^v ^v ^v ^v livre un tas de cadeaux à des enfants très chanceux dans seulement 2 maisons.

4.3.2 seconde partie

L'année suivante, pour accélérer le processus, le Père Noël crée une version robotisée de lui-même, Robo-Santa, pour livrer les cadeaux avec lui.

Le Père Noël et Robo-Santa commencent au même endroit (ils livrent deux cadeaux à la même maison de départ), puis se déplacent à tour de rôle en fonction des instructions du lutin, qui lit dans le lait de poule le même scénario que l'année précédente.

Cette année, combien de maisons recevront au moins un cadeau ?

Par exemple :

^v livre des cadeaux à 3 maisons, parce que le Père Noël va au nord et que Robo-Santa va au sud.

^ >v < livre maintenant des cadeaux à 3 maisons, et le Père Noël et Robo-Santa reviennent à leur point de départ.

^v ^v ^v ^v ^v livre maintenant des cadeaux à 11 maisons, le Père Noël allant dans une direction et Robo-Santa dans l'autre.

4.4 Exercice 4. Année 2015, jour 5

Vous avez en entrée un fichier texte `exo4.txt`. En sortie, vous devrez afficher la valeur d'une variable sur votre terminal de commande.

4.4.1 première partie

— Jour 5 : Il n'a pas de stagiaires pour cela ? —

Le Père Noël a besoin d'aide pour déterminer quelles chaînes de son fichier texte sont bonnes ou fausses.

Une chaîne de caractères bonne est une chaîne qui possède toutes les propriétés suivantes :

Elle contient au moins trois voyelles (aeiou uniquement), comme `aei`, `xazegov`, ou `aeiouaeiouaeiou`.

Elle contient au moins une lettre qui apparaît deux fois de suite, comme `xx`, `abcdde` (`dd`), ou `aabbccdd` (`aa`, `bb`, `cc`, ou `dd`). Elle ne contient pas les chaînes `ab`, `cd`, `pq` ou `xy`, même si elles font partie de l'une des autres exigences.

Par exemple :

`ugknbfddgicrmopn` est bonne parce qu'elle contient au moins trois voyelles (`u...i...o...`), une lettre double (`...dd...`), et aucune des sous-chaînes interdites. `aaa` est bonne parce qu'elle contient au moins trois voyelles et une lettre double, même si les lettres utilisées par les différentes règles se chevauchent.

`jchzalrnumimnmhp` est fausse car elle n'a pas de lettre double.

`haegwjzuvuyypxyu` est fausse car elle contient la chaîne `xy`.

`dvszwmarrgswjxmb` est fausse car elle ne contient qu'une seule voyelle.

Combien de chaînes sont bonnes ?

4.4.2 seconde partie

Conscient de son erreur, le Père Noël a adopté un meilleur modèle pour déterminer si une ficelle est bonne ou fausse. Aucune des anciennes règles ne s'applique, car elles sont toutes clairement ridicules.

Désormais, une chaîne de caractères bonne est une chaîne qui possède toutes les propriétés suivantes :

Elle contient une paire de deux lettres quelconques qui apparaissent au moins deux fois dans la chaîne sans se chevaucher, comme xyxy (xy) ou aabcdefgaa (aa), mais pas comme aaa (aa, mais elle se chevauche). Elle contient au moins une lettre qui se répète avec exactement une lettre entre elles, comme xyx, abcdefeghi (efe), ou même aaa.

Par exemple :

qjvhvtzxzqjkmph est bonne parce qu'elle contient une paire qui apparaît deux fois (qj) et une lettre qui se répète avec exactement une lettre entre elles (zxz).

xyyxx est bonne parce qu'elle contient une paire qui apparaît deux fois et une lettre qui se répète avec une lettre entre les deux, même si les lettres utilisées par chaque règle se chevauchent.

uurcxstgmygtbstg est fausse car elle a une paire (tg) mais pas de répétition avec une seule lettre entre les deux.

ieodomkazucvgmuy est fausse parce qu'elle contient une lettre répétée avec une lettre entre les deux (odo), mais pas de paire apparaissant deux fois.

Combien de chaînes sont bonnes selon ces nouvelles règles ?

5 Corrections

5.1 TP 1

Sortie terminal

```
1 print("Hello \n ceci est un programme \n  en python !");
```

Entrée utilisateur

```
1 """
2 Définition: Programme qui demande à l'utilisateur son nom et affiche ce nom sur le terminal.
3 réalisé dans le cadre du module R1.03 Bases de la programmation
4 Auteur: Mme Tabary
5 Date: 2/08/2023    Version: 1.0
6 """
7
8 #affichage sur le terminal
9 print("Bonjour, quel est ton nom ?")
10 #saisir une valeur au clavier avec input et affectation de la variable nom
11 nom = input()
12
13 #affichage sur le terminal
14 print("quel est ton âge ?")
15 #saisir une valeur au clavier avec input et affectation de la variable age
16 age = input()
17
18 #affichage sur le terminal
19 print("quel est ta taille ?")
20 #saisir une valeur au clavier avec input et affectation de la variable taille
21 taille = input()
22
23
24 ANNEE = 2023
25 # affichage du contenu de la variable nom
26 print("Bonjour humain au nom de", nom, ", âgé de ", age, " ans, né en ", ANNEE - int(age),
27       "et mesurant", taille[:1], "m", taille[1:3], "cm.")
```

Boucle for

```
1 """
2 Définition: Programme qui demande à l'utilisateur son nom et affiche ce nom sur le terminal.
3 réalisé dans le cadre du module R1.03 Bases de la programmation
4 Auteur: Mme Tabary
5 Date: 3/08/2023    Version: 1.0
6 """
7
8 #tableau de la classe
9 listeEtudiants = ["Alice", "Bob", "Charlie", "Diane", "Alice", "Alice", "Charlie", "Charlie"]
10
11 #affichage des étudiants présents
12 dup = {x for x in listeEtudiants if listeEtudiants.count(x) > 1}
13 print(dup)
```

Boucle tant que

```
1 """
```

```

2  Définition: Programme qui demande à l'utilisateur son nom et affiche ce nom sur le terminal.
3  réalisé dans le cadre du module R1.03 Bases de la programmation
4  Auteur: Mme Tabary
5  Date: 3/08/2023    Version: 1.0
6  """
7
8  #tableau de la classe
9  listeEtudiants = ["Alice", "Bob", "Charlie", "Diane"]
10
11
12
13 #verification de la présence des étudiants
14 for w in listeEtudiants:
15     print("L'étudiant ",w, "est-il venu en cours ?")
16     reponse = input()
17     #verification de la réponse
18     while reponse != "y" and reponse != "n":
19         print("reponse incorrecte, veuillez saisir y ou n")
20         reponse = input()
21     if reponse == "y":
22         print("L'étudiant ",w, "est-il bien orthographié ?")
23         reponse = input()
24         #verification de la réponse
25         while reponse != "y" and reponse != "n":
26             print("reponse incorrecte, veuillez saisir y ou n")
27             reponse = input()
28         if reponse == "n":
29             #modification de l'orthographe
30             print("Veuillez ressaisir le nom de l'étudiant")
31             nouveauNom = input()
32             #verification de la réponse
33             print("souhaitez-vous remplacer ", w, " par ", nouveauNom, " ?")
34             reponseOrthographie = input()
35             #boucle de vérification de la réponse
36             while reponseOrthographie == "n":
37                 print("Veuillez ressaisir le nom de l'étudiant")
38                 nouveauNom = input()
39                 print("souhaitez-vous remplacer ", w, " par ", nouveauNom, " ?")
40                 reponseOrthographie = input()
41             #remplacement de l'ancien nom par le nouveau dans la liste
42             listeEtudiants = list(map(lambda x: x.replace(w, nouveauNom), listeEtudiants))
43
44 #affichage des étudiants présents
45 print(listeEtudiants)

```

Juste prix

```

1  """
2  Définition: Programme qui demande à l'utilisateur son nom et affiche ce nom sur le terminal.
3  réalisé dans le cadre du module R1.03 Bases de la programmation
4  Auteur: Mme Tabary
5  Date: 3/08/2023    Version: 1.0
6  """
7
8  #importation de la librairie random

```

```

9 import random
10
11 #affectation d'un nombre aléatoire entre 1 et 10 à la variable prix
12 prix = random.randint(1, 10)
13
14 #initialisation de la variable score
15 SCORE = 100
16
17 #initialisation de la variable tentatives
18 tentatives = 0
19
20
21 print("Devinez le juste prix ! Le prix est un nombre compris entre 1 et 10 inclus.")
22
23 #boucle de jeu
24 while True:
25     nombre = int(input())
26     tentatives += 1
27     if nombre < prix:
28         print("Le just prix est plus haut")
29     if nombre > prix:
30         print("Le juste prix est plus bas")
31     if nombre == prix:
32         print("Félicitations, vous avez trouvé le juste prix {} en {} essais, votre score est {} !".format
33             (prix, tentatives, int(score / tentatives)))
34         break
35 print("Partie terminée")

```

5.2 TP 2

liste

```

1
2 """
3 Définition: Programme qui demande à l'utilisateur son nom et affiche ce nom sur le terminal.
4 réalisé dans le cadre du module R1.03 Bases de la programmation
5 Auteur: Mme Tabary
6 Date: 3/08/2023    Version: 1.0
7 """
8
9 import random
10
11 #affectation d'une liste
12 listeTout = ['chat', 1, 'chien', 3, 21,
13             'tortue', 1, True, False,
14             5, 5.5, 'b'
15             ]
16 listeEntiers = []
17 listeChaine = []
18 listeBool = []
19 listeFloat = []
20 listeAutre = []
21

```

```

22 #affichage de la liste
23 print("la liste est :", \
24     listeTout)
25
26 #affectation de la longueur de la liste à la variable longueurListeTout
27 longueurListeTout = len(listeTout)
28
29 #boucle de parcours de la liste
30 for i in range(longueurListeTout):
31     if type(listeTout[i]) == int:
32         print("c'est un entier")
33         listeEntiers.append(listeTout[i])
34     elif type(listeTout[i]) == str:
35         print("c'est une chaîne de caractère")
36         listeChaine.append(listeTout[i])
37     elif type(listeTout[i]) == bool:
38         print("c'est un booléen")
39         listeBool.append(listeTout[i])
40     elif type(listeTout[i]) == float:
41         print("c'est un float")
42         listeFloat.append(listeTout[i])
43     else:
44         print("c'est autre chose")
45         listeAutre.append(listeTout[i])
46
47 listeEntiersSorted = sorted(listeEntiers)
48 listeChaineReverse = listeChaine[::-1]
49
50
51 #affichage des nouvelles listes
52 print("les nouvelles listes sont :", \
53     listeEntiersSorted, listeBool, listeChaineReverse, listeFloat, listeAutre)
54
55 #affectation de la nouvelle liste
56 maNouvelleListe = listeEntiersSorted[-2:-1] + listeBool[-2:-1] + \
57     listeChaineReverse[-2:-1] + listeFloat[-2:-1] + listeAutre[-2:-1]
58 #affichage de la nouvelle liste
59 print("ma nouvelle liste est :", \
60     maNouvelleListe)
61
62 #affichage d'une valeur aléatoire de la nouvelle liste
63 print("la valeur aléatoire est :", random.choice(maNouvelleListe))

```

string

```

1
2 """
3 Définition: Programme qui demande à l'utilisateur son nom et affiche ce nom sur le terminal.
4 réalisé dans le cadre du module R1.03 Bases de la programmation
5 Auteur: Mme Tabary
6 Date: 3/08/2023    Version: 1.0
7 """
8
9
10 #affectation d'une liste

```

```

11 listeTest = ['J\'étudie dans la formation du','BUT', 'Sciences', 'des', 'Données' ]
12
13 #création d'une chaine de caractère à partir d'une liste
14 monString = " ".join(listeTest)
15
16 print(monString)
17
18 #découpage de la chaine de caractère en liste
19 listeFinale = monString.split()
20
21 #affichage de la liste
22 print("la liste est :", listeFinale)
23
24 #affichez le nombre de mots de la liste
25 print("le nombre de mots de la liste est :", len(listeFinale))
26
27 #affichez le nombre 'd'occurrences du mot "BUT"
28 print("le nombre 'd'occurrences du mot \"BUT\" est :", listeFinale.count("BUT"))
29
30 #rajouter entre "J'étudie" et "dans" les éléments "à", "Dole"
31 listeFinale.insert(1, "à")
32 listeFinale.insert(2, "Dole")
33 print("la liste est :", listeFinale)
34
35 #supprimer le mot "Données"
36 listeFinale.remove("Données")
37
38 #supprimer le mot "des"
39 listeFinale.remove("des")
40
41 monStringFinal = " euh ".join(listeFinale)
42 print("le string devient :", monStringFinal)
43
44 #afficher le string en majuscule
45 monStringFinalMaj = monStringFinal.upper()
46 print("le string en majuscule est :", monStringFinalMaj)
47
48 #remplacez les mots "euh" par des espaces
49 monStringFinalMajSansEUH = monStringFinalMaj.replace("EUH", " ")
50
51 print("le string sans euh est :", monStringFinalMajSansEUH)
52
53 #vérifiez si le mot "Dole" est dans le string
54 print("le mot \"Dole\" est-il dans le string ? :", "DOLE" in monStringFinalMajSansEUH)
55
56 #affichez le string en minuscule
57 monStringFinalMajSansEUHlower = monStringFinalMajSansEUH.lower()
58 print("le string en minuscule est :", monStringFinalMajSansEUHlower)
59 print(monStringFinalMajSansEUHlower.split())

```

tuple

```

1
2 """
3 Définition: Programme de manipulation de tuples.

```

```

4  réalisé dans le cadre du module R1.03 Bases de la programmation
5  Auteur: Mme Tabary
6  Date: 3/08/2023    Version: 1.0
7  """
8
9  #affectation d'un tuple
10 depensesTuple = (5, 21, 101, 423)
11
12 #affectation de variables
13 (gaz, forfait, course, loyer) = depensesTuple
14
15 #affichage des variables
16 print("le tuple est :")
17 print(course)
18 print(forfait)
19 print(loyer)
20
21 #affectation tuple sur 3 mois
22 mesPrevisionsTroisMois = depensesTuple * 3
23
24 #affichage du nouveau tuple
25 print(mesPrevisionsTroisMois)

```

set

Instructions	Sortie écran
$s = \{3, 4, \text{"Plouf"}\}, (1, 3)$ <code>print(s)</code>	$\{\text{'Plouf'}, (1, 3), 3, 4\}$
$s2 = \{3.14, [1, 2]\}$ <code>print(s)</code>	Erreur : liste non hashable
<code>print((set((2, 2, 2, 1))))</code>	$\{1, 2\}$
$s3 = \text{set}(\text{"BUTSDS1"})$ <code>print(s3)</code>	$\{\text{'T'}, \text{'D'}, \text{'B'}, \text{'S'}, \text{' '}, \text{'1'}, \text{'U'}\}$
<code>s3.add(1)</code> <code>print(s3)</code>	$\{1, \text{'T'}, \text{'D'}, \text{'B'}, \text{'S'}, \text{' '}, \text{'U'}, \text{'1'}\}$
<code>s3.discard('1')</code> <code>print(s3)</code>	$\{1, \text{'T'}, \text{'D'}, \text{'B'}, \text{'S'}, \text{' '}, \text{'U'}\}$
<code>s3.discard('t')</code> <code>print(s3)</code>	$\{1, \text{'T'}, \text{'D'}, \text{'B'}, \text{'S'}, \text{' '}, \text{'U'}\}$
<code>print(s3.intersection(s))</code>	<code>set()</code>

dictionnaires

```

1  #affectation d'un dictionnaire
2  thisdict = {
3      "nom": "Tabary",
4      "profession": {"Enseignant", "chercheur", "Directice d'études", "Maitre de conférences"},
5      "age": 1988
6  }
7
8  #affichage du dictionnaire
9  print("le disctionnaire est ", thisdict)
10

```

```

11 #affichage du dictionnaire clé par clé
12 for key in thisdict:
13     print(key, "(cle) : ", thisdict[key], "(valeur)")
14
15 #vérification de l'existence d'une clé
16 if "age" in thisdict:
17     print("La clé 'age' existe pour thisdict")
18
19 #affichage du dictionnaire ordonné
20 print (sorted(thisdict))
21
22 #affectation d'un dictionnaire
23 thisdict2 = {
24     "nom": "Mohammed",
25     "profession": {"Développeur web", "Enseignant", "Manager"},
26     "age": 1987
27 }
28
29 #liste de dictionnaires
30 mesProfs = [thisdict, thisdict2]
31
32 monEnsemble = set(thisdict["profession"])
33
34 for item in mesProfs:
35     for key in item:
36         if key == "profession" :
37             monEnsemble = monEnsemble & item[key]
38
39 print (monEnsemble)

```

projet

```

1 element = 5
2 liste_triee =[1,2,3,4,5,6,7,8,9,10]
3 a = 0
4 b = len(liste_triee)-1
5 m = (a+b)//2
6 while a < b :
7     if liste_triee[m] == element:
8         print("reponse : ", m)
9         break
10    elif liste_triee[m] > element:
11        b = m-1
12    else :
13        a = m+1
14    m = (a+b)//2

```

5.3 TP 3

avec un fichier texte

```

1 """
2 Définition: Programme qui concatène les fichiers bilbo1.txt, bilbo2.txt et bilbo3.txt 7
3 dans un nouveau fichier bilbo.txt
4 et qui remplace hobbit par Periannath dans le fichier bilbo.txt

```

```

5  réalisé dans le cadre du module R1.03 Bases de la programmation
6  Auteur: Mme Tabary
7  Date: 7/08/2023    Version: 1.0
8  """
9
10 # Concaténation de fichiers
11 # ouverture du fichier en lecture
12 bilbo1 = open("fichiersTexte/bilbo1.txt", "r")
13 bilbo2 = open("fichiersTexte/bilbo2.txt", "r")
14 bilbo3 = open("fichiersTexte/bilbo3.txt", "r")
15 bilbo = open("fichiersTexte/bilbo.txt", "w")
16
17 # Ecriture dans le fichier
18 bilbo.write(bilbo1.read()+bilbo2.read()+bilbo3.read())
19
20 # Fermeture des fichiers (important)
21 bilbo.close()
22 bilbo1.close()
23 bilbo2.close()
24 bilbo3.close()
25
26
27 # Vérification du fichier créé
28 bilbo = open("fichiersTexte/bilbo.txt", "r")
29 print("Le hobbit : \n", bilbo.read())
30 bilbo.close()
31
32
33 # Remplacement de hobbit par Periannath
34 with open("fichiersTexte/bilbo.txt", "r") as bilbo, open("fichiersTexte/bilboE.txt", "w") as bilboE:
35     for ligne in bilbo:
36         bilboE.write(ligne.replace("hobbit", "Periannath"))
37
38 # Vérification du fichier créé
39 bilboE = open("fichiersTexte/bilboE.txt", "r")
40 print("\n Version elfique du hobbit : \n", bilboE.read())
41 bilboE.close()

```

5.4 TP 4

La réponse à l'exo1, première partie, est 232. La réponse à l'exo1, seconde partie, est 1783.

La réponse à l'exo2, première partie, est 1598415. La réponse à l'exo2, seconde partie, est 3812909.

La réponse à l'exo3, première partie, est 2592. La réponse à l'exo3, seconde partie, est 2360.

La réponse à l'exo4, première partie, est 238. La réponse à l'exo4, seconde partie, est 2360.