Utilisation de l’IA pour apprendre **PYTHON**

version 1.0 par Christophe Dabancourt christophe.dabancourt@univ-evry.fr

L’étudiant est autonome

Le cours est progressif

Toutes les questions peuvent être posées à l’IA

* https://claude.ai/new
* https://llama3.dev/
* https://chatgpt.com/

rq : le plus simple, est de se connecter avec un compte (gmail par exemple)

pour info, j’ai demandé à l’IA : « je suis enseignant en informatique, et je veux enseigner le python à des étudiants qui n'ont jamais programmer. Et je veux qu'ils utilisent l’IA pour apprendre le python. quel plan proposes tu sur 10 séances de 2h. »

rq : tous les prompt de ce cours seront surlignés en vert

Voici la réponse :

*Pour enseigner le Python à des étudiants sans expérience préalable en programmation et en intégrant l'utilisation de ChatGPT comme outil d'apprentissage, voici un plan de cours sur 10 séances de 2 heures chacune. L'objectif est de leur donner une base solide en Python, les initier aux concepts fondamentaux de la programmation et les rendre autonomes dans l'apprentissage et la résolution de problèmes.*

Bien sur, j’ai modifier quelque peu l’ordre des chapitres et j’ai ajouté la création d’une IHM (fenêtre graphique avec des boutons et des images) pour faire faire des applications ludiques qui encouragent les étudiants à aller plus loin...

Chaque partie peut-être faite en autonomie.

L’étudiant doit lire la consigne.

Pour chaque « Exercice », dans cet ordre :

1. L’étudiant recopie le sujet dans l’interface de l’IA
2. L’étudiant recopie le code généré par l’IA pour le tester
3. L’étudiant doit lire les explications données par l’IA

et si l’étudiant a une question... il doit la poser à l’IA (et lire la réponse)

Objectif du cours :

- apprendre la programmation

- lire un code (notion de variables, boucles, fonctions...)

- tester un code (pratique)

- savoir ce que peut faire un programme (structures, fonctions, types...)

- apprendre à écrire (et améliorer) des prompts

- savoir ce qu’on veut obtenir

- savoir intégrer des bouts de code dans son programme

Je vous conseille de créer un fichier texte en parallèle du programme à écrire (avec un traitement de texte par exemple) pour sauvegarder l’ensemble de vos prompts.

**Pré-requis**: « avant de copier des instructions, il faut connaître le minimum »

Un programme se compose d’une succession d’instructions.

Toutes les instructions s’exécutent de la 1ere ligne à la dernière (du haut vers le bas) l’une après l’autre.

Le programme utilise des variables pour stocker des valeurs.

Une variable est une case mémoire qui contient **une valeur**.

On identifie une variable par **un nom**

On dit qu’une variable à **un type** :

* type nombre entier : une\_variable = 48
* type nombre réel : pi = 3.14
* type string (chaîne de caractères) mot = "guitare"

Vous pouvez utiliser autant de variables que nécessaire (avec un nom différent pour chaque variable).

Voici un 1er programme de 5 lignes :

ligne 1 : print( "Bonjour !" ) # affiche le mot « Bonjour !  »

ligne 2 : temp = 0 # on affecte la valeur 0 à la variable temps

ligne 3 : temp = temp + 5 # on calcule d’abord la partie droite : temp +5

# la valeur de temp (défini avant) est égale à 0

# donc temp + 5 vaut la valeur 5

# donc on fait « temp = 5 » :

# la variable temp vaut 5 après la ligne n°3

ligne 4 : temp = 20 – 2 \* temp # on calcule la partie droite (avec temp qui vaut 5)

# « 20 – 2 \* temp » vaut « 20 – 2 \* 5 » vaut 10

ligne 5 : print(temp)

le programme précédent affichera toujours les 2 lignes suivantes :

Bonjour !

10

Un programme peut faire plusieurs milliers de lignes.

La valeur d’une variable ne change pas, sauf si une instruction le fait explicitement.

### Séance 1 : Introduction à la programmation et à Python

1. \*\*Objectifs\*\* : Comprendre les concepts de base de la programmation et l'environnement Python.

2. \*\*Contenu\*\* : pour installer et tester PYTHON, le mieux est de suivre une vidéo...

**- Installation de Python et d'un IDE (par exemple, un éditeur de texte, PyCharm, VS Code ou Thonny).**

**- Premier programme Python : "Bonjour !"**

\*\*Objectifs\*\* : Comprendre les types de données, les variables et les opérations de base.

3. \*\*Contenu\*\* :

- Types de données : int, float, str, bool.

- Variables et affectations.

- Opérations arithmétiques et logiques.

4. \*\*Exercices\*\* :

- Manipulation des variables et des types de données ( int, float, str, bool )

- Petits calculs et affichages.

- Utiliser ChatGPT pour explorer des fonctions intégrées et résoudre des exercices simples :

Exercice 1 : Je veux écrire un programme simple en python. Le programme doit demander à l'utilisateur d'entrer son prénom et son nom, puis affichez un message de bienvenue.

Exercice 2 : Écrivez un programme qui demande une phrase à l'utilisateur et affiche le nombre de caractères dans cette phrase.

Exercice 3 : Demandez à l'utilisateur d'entrer deux nombres entiers, puis affichez la somme, la différence, le produit et le quotient de ces nombres.

Exercice 4 : Demandez à l'utilisateur d'entrer la base et la hauteur d'un triangle, puis calculez et affichez la surface du triangle.

Exercice 5 : Écrivez un programme qui convertit une température de Celsius en Fahrenheit.

Exercice 6 : Écrivez un programme qui vérifie si un utilisateur est majeur (18 ans ou plus).

Exercice 7 : Demandez à l'utilisateur d'entrer deux nombres, puis vérifiez et affichez si le premier nombre est supérieur au deuxième.

Découverte : «**input**», «**print**», «  **int (**input( ) **)** », «  **float (**input ( ) **)** »,

«  **f"** var=**{**var**} "** »

4.8 Exercice 8 : Vous avez une recette de gâteau pour 4 personnes avec les proportions suivantes :

* 200 g de farine
* 150 g de sucre
* 100 ml de lait
* 100 g de beurre
* 2 œufs

Votre tâche est de créer un programme en Python qui demande à l'utilisateur le nombre de personnes pour lesquelles il souhaite préparer le gâteau, puis affiche les quantités nécessaires pour chaque ingrédient en fonction du nombre de personnes. Vous ferez attention que le nombre d’œufs est un nombre entier (arrondi au nombre supérieur). Sans utiliser les listes ni les dictionnaires.

### Séance 2 : Les structures de contrôle : CONDITIONS et BOUCLES

1. \*\*Objectifs\*\* : Comprendre et utiliser les structures de contrôle pour diriger le flux du programme.

2. \*\*Contenu\*\* :

- Les conditions : if, elif, else.

- Les boucles : for et while.

- Les opérateurs de comparaison et logiques.

3. \*\*Exercices\*\* :

**- Écrire des programmes utilisant des conditions.**

- Utiliser ChatGPT pour des questions sur les erreurs courantes et les conseils de débogage.

Exercice 0 : Demandez à l’IA la question suivante (lire et testez la réponse)

comment expliquer l'usage des tabulations et des blocs dans un programme python à travers 3 exemples simples

Exercice 1 : Écrivez un programme qui vérifie si un utilisateur est majeur (18 ans ou plus). Il doit saisir son âge.

Exercice 2 : Écrivez un programme qui demande à l'utilisateur son âge et affiche une catégorie en fonction de cet âge :

* Moins de 12 ans : "Enfant"
* Entre 12 et 17 ans : "Adolescent"
* Entre 18 et 64 ans : "Adulte"
* 65 ans et plus : "Senior"

Exercice 3 : Calculateur d'impôt simplifié. Écrivez un programme qui calcule l'impôt sur le revenu en fonction des tranches d'imposition suivantes :

* Revenu jusqu'à 10 000 € : 0%
* Revenu entre 10 001 € et 25 000 € : 10%
* Revenu entre 25 001 € et 50 000 € : 20%
* Revenu au-delà de 50 000 € : 30%

Le programme doit demander le revenu annuel de l'utilisateur et afficher le montant de l'impôt à payer.

Exercice 4 : Écrivez un programme qui demande à l'utilisateur d'entrer une date sous la forme de trois entiers (jour, mois, année). Le programme doit vérifier si la date est valide. Voici les règles à suivre :

1. Les jours doivent être compris entre 1 et 31.
2. Les mois doivent être compris entre 1 et 12.
3. Février a 28 jours, sauf si l'année est bissextile où il a 29 jours.
4. Les mois d'avril, juin, septembre et novembre ont 30 jours.
5. Les autres mois ont 31 jours.

Une année est bissextile si elle est divisible par 4 mais pas par 100, sauf si elle est aussi divisible par 400.

Découverte : «**if**», «**elif**», «**else**», «  **==**  », «  **!=**  »,, «  <**=**  », «  < », «  **or** » , «  **and** »

et l’utilisation des blocs de programme (avec les tabulations)

Pourquoi les tabulation (ou 4 espaces) pour délimiter des indentations / système de blocs ?

Voici un programme simple :

Question 1 : Le « else » est-il associé au 1er « if » ou au 2ème ?

Question 2 : Ce programme peut-il fonctionner

age=int(input("entrez votre age"))

if (age<=18) :

print("mineur")

if (age>60) :

print("senior")

else :

print("faites du sport")

Ce programme ne peut pas être exécuté car l’interpréteur ne sait pas ce que veut faire le développeur… on ne peut pas dire à quel « if » est associé le « else », c’est pourquoi il y a des blocs/indentations. Les 3 programmes suivants sont différents :

|  |  |  |
| --- | --- | --- |
| if (age<=18) :  print("mineur")  if (age>60) :  print("senior")  else :  print("faites du sport") | if (age<=18) :  print("mineur")  if (age>60) :  print("senior")  else :  print("faites du sport") | if (age<=18) :  print("mineur")  if (age>60) :  print("senior")  else :  print("faites du sport") |

Si besoin, demandez plus d’explications à l’IA

en python, peux-tu m'expliquer l'utilité des indentations

ou voir le site suivant :

https://koor.fr/Python/Tutorial/python\_instruction\_bloc.wp

------------------------

4. \*\*Exercices\*\* : Les BOUCLES

**- Créer des boucles pour des tâches répétitives.**

- Utiliser ChatGPT pour des questions sur les erreurs courantes et les conseils de débogage.

Exercice 5 : Écrivez un programme qui utilise une boucle «**for**» pour afficher les nombres de 1 à 10.

Exercice 6 : Écrivez un programme qui utilise une boucle « **while** » pour afficher les nombres de 1 à 10.

Exercice 7 : Écrivez un programme qui affiche tous les nombres multiples de 11 compris entre 1 et 1000.

Exercice 8 : Écrivez un programme qui demande à l'utilisateur d'entrer un nombre entre 0 et 20. Si l’utilisateur fait une erreur, le programme demande à nouveau le nombre.

Variante : Écrivez un programme qui demande à l'utilisateur d'entrer un nombre entre 0 et 20. Si l’utilisateur fait une erreur, le programme demande à nouveau le nombre. Au bout de 5 essais, le programme affiche « erreur de saisie » et sort de la boucle.

Exercice 9 : Écrivez un programme qui affiche les tables de multiplication de 1 à 10 sous forme de tableau. Utilisez des boucles for imbriquées pour générer et afficher les valeurs correctement formatées.

variante : Écrivez le même programme en utilisant des boucles « while »

Exercice 10 : Écrivez un programme qui utilise une boucle « for » pour afficher les nombres de 10 à 1.

variante : Écrivez le même programme en utilisant des boucles « while »

Exercice 11 : Demandez à l’utilisateur de saisir un nombre entier positif N. Et calculer la somme des N premiers entiers ( 1+2+.. + N).

variable : afficher la valeur de la variable Nx(N+1)/2. Qu’en déduisez- vous ?

Découverte : «**range**», « **for** », «**while**», et les « blocs de programme (avec la tabulation) »

Exercice 11 : Voici 2 programmes absolument identiques, écrit différemment : (testez les pour vérifier)

|  |  |
| --- | --- |
| temp=0  for i in range(3) :  temp = temp +2  print( temp ) | temp = 0  i = 0  temp = temp +2  i = 1  temp = temp +2  i = 2  temp = temp +2  print( temp ) |
| Une instruction dans une boucle (3 fois) | On écrit 3 fois la même instruction |

PROJET et CONTRÔLE

**PROJET** (à lire, et à faire faire par l’IA bien sur)

Vous devez créer un programme python (sans utiliser de liste ni de dictionnaire) afin de déterminer le prix de location d’un véhicule. Ce prix dépend de la catégorie du véhicule, de nombre de jours de location, des options prises par le client et de l’assurance optionnelle proposée.

Le loueur propose 4 catégories de véhicules :

Code de la catégorie Nom de la catégorie Prix de location par jour

A Compact 50€

B Berline 70€

C SUV 80€

D Prestige 150€

Les options proposées par le loueur sont les suivantes :

• Le GPS intégré : 8€ par jour

• Des sièges « bébé » : 5€ par jour et par siège

• L’assurance « suppression franchise » : 19€ par jour

• Deuxième conducteur : +15% du prix de location (hors options)

Pour chaque option, l’utilisateur répond « o » pour « oui » ou « n » pour « non »

attention:quand vous demandez « voulez-vous un siège bébé?(o/n) »

si vous répondez « o », il faut demander combien de sièges bébé.

**CONTRÔLE**

(cherchez 5 minutes **sans machine**, puis copiez/collez le programme pour afficher la solution)

**Voici un programme. Identifier les valeurs des variables aux étapes n°1, n°2 et n°3**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| temp = 5 |  |  |  |  |
| i = 0 |  |  |  |  |
| temp = temp + 1 |  |  |  |  |
| temp = 2 \* i - temp |  |  |  |  |
| if (temp < 0) : |  |  |  |  |
| nb = 1 |  | temp | i | nb |
| elif (i==0) : | étape N°1 |  |  |  |
| nb = 2 | étape N°2 |  |  |  |
| else : | étape N°3 |  |  |  |
| nb = 3 |  |  |  |  |
| #-------------------- étape n°1 |  |  |  |  |
| for nb in range (3) : |  |  |  |  |
| for col in range(5) : |  |  |  |  |
| temp = temp + 2 |  |  |  |  |
| #-------------------- étape n°2 |  |  |  |  |
| i = 0 |  |  |  |  |
| while i < 4 : |  |  |  |  |
| print(i) |  |  |  |  |
| temp = i |  |  |  |  |
| i=i+1 |  |  |  |  |
| #-------------------- étape n°3 |  |  |  |  |

### Séance 4 : Les fonctions

1. \*\*Objectifs\*\* : Apprendre à créer et utiliser des fonctions pour organiser le code.

2. \*\*Contenu\*\* :

- Définition et appel de fonctions.

- Paramètres et arguments.

- Valeurs de retour.

- Portée des variables.

3. \*\*Exercices\*\* :

- Écrire des fonctions simples.

- Utiliser ChatGPT pour comprendre les erreurs de fonctions et améliorer les solutions.

- Projets simples utilisant des fonctions.

Création de fonction personnalisée :

Exercice 1 : Écrivez une fonction appelée « *saluer »* qui prend un argument nom et affiche une salutation personnalisée. Vous testerez cette fonction avec 4 appels.

Exercice 2 : Écrivez une fonction qui appelée « *fois2 »* qui prend un argument nombre et retourne la valeur du nombre fois deux (doublée). Vous testerez l’appel à cette fonction avec tous les nombre de 1 à 20 dans une boucle « for ».

Exercice 3 : Écrivez une fonction qui appelée « *contient »* qui prend un argument une chaine de caractères (une phrase) et retourne le nombre fois que cette chaîne contient la lettre « a ». Cette fonction ne doit pas utiliser la méthode « count ». Vous testerez l’appel à cette fonction en demandant à l’utilisateur de saisir une chaîne de caractère.

Exercice 4 : Écrivez une fonction appelée « *somme\_chiffres »* qui prend un nombre entier positif en entrée et renvoie la somme de ses chiffres.

Exemple : somme\_chiffres(123) -> 1 + 2 + 3 = 6 et somme\_chiffres(456) -> 4 + 5 + 6 = 15

Exercice 5 : Écrivez une fonction appelée « *plus\_petit »* qui prend 3 nombres en entrée et renvoie le plus petit (sans utiliser la fonction min).

Exercice 6 : Écrivez un exemple de fonction en donnant des valeurs par défaut aux paramètres avec des exemples d’appels. Comme ça, si lors de l’appel de la fonction, on ne passe pas le paramètre, la fonction s’exécutera en prenant la valeur définit par défaut pour ce paramètre manquant.

Découverte : fonctions avec 1, 2 ou 3 paramètres. « **%**», « **//** »,

«**def nom\_fonction(parametres) :**», les « blocs de programme (avec la tabulation) » ,

« appel d’une fonction » , « for i in chaine :  »

« paramètres par défaut »

Utilisation de fonctions qui existent déjà :

les fonctions sur les nombres

Exercice 7 : Écrivez un programme qui demande à l’utilisateur de saisir 3 nombres, puis affiche le plus petit (utilisation de « min »), le plus grand (max), la somme (sum), la valeur absolue du plus petit moins le plus grand.

Affiche le 1er nombre divisé par 11 arrondit à 2 décimales.

Exercice 8 : que se passe-t-il quand on a une variable a = 0.1 + 0.1 + 0.1 ? expliquez.

Exercice 9 : Afficher une fonction sinus et cosinus

Exercice 10 : Dessiner une fonction sinus et cosinus

les fonctions sur les chaînes de caractères

Exercice 11 : Écrivez un programme qui utilise les fonctions suivantes :

* len
* lower
* upper
* strip
* split
* replace
* find

Découverte : les fonctions pour les « string » s’utilisent différemment. On place la variable, suivi d’un point, suivi du nom de la fonction…

rq : on verra ça plus tard. Dit rapidement, on considère que la variable est un « objet », et chaque type d’objet possède certaines fonctions particulières (appelées « méthodes »)

### Séance 5 : Les listes et les tuples

1. \*\*Objectifs\*\* : Comprendre et manipuler les collections de données.

2. \*\*Contenu\*\* :

- Les listes : création, manipulation, méthodes.

- Les tuples : caractéristiques et utilisations.

- Indexation et slicing.

3. \*\*Exercices\*\* :

- Créer et manipuler des listes et des tuples.

- Utiliser des boucles pour parcourir des collections.

- Utiliser ChatGPT pour explorer des méthodes de liste et résoudre des exercices de manipulation de données.

Exercice 1 : Demandez à l’utilisateur de saisir une liste de mots. La saisie se termine quand l’utilisateur saisit « q » pour quitter.

* Afficher la liste saisie.
* Afficher le 2ème élément de la liste
* Afficher la liste en sens Inverse
* Afficher le mot de la liste qui a le plus de lettres
* Supprimer les doublons dans la liste

Exercice 2 : Demandez à l’utilisateur de saisir une liste de nombres. La saisie se termine quand l’utilisateur saisit « 0 » pour quitter.

* Parcourir la liste pour calculer et afficher la somme (sans utiliser la fonction « sum »)
* parcourir la liste pour identifier le plus petit élément (sans utiliser la fonction « min »)
* enlever le plus petit élément de la liste
* afficher la liste

variante : la même chose sans utiliser les fonctions « sum » et « min »

Exercice 3 : Demandez à l’utilisateur de saisir une liste de nombres et de chaines. La saisie se termine quand l’utilisateur saisit « q » pour quitter. Afficher un élément sur deux de la liste (le 1er, le 3eme...)

variante : afficher le nombre d’éléments qui sont des nombres, et le nombre d’élements qui sont des string.

Découverte : créer une variable de type « liste », ajouter des éléments avec « append »

les enlever avec « remove ».

Parcourir une liste avec « for ».

connaître le type d’une variable avec « isinstance »

1. \*\*Objectifs\*\* : Comprendre et manipuler les collections de données.

2. \*\*Contenu\*\* :

- Les tuples : caractéristiques et utilisations.

- Indexation et slicing.

3. \*\*Exercices\*\* :

**- Créer et manipuler des tuples.**

- Utiliser des boucles pour parcourir des collections.

- Utiliser ChatGPT pour explorer des méthodes de liste et résoudre des exercices de manipulation de données.

Exercice 1 : c'est quoi un tuple (avec 3 exemples)

Exercice 2 : quelle est la différence en un tuple et une liste ?

Exercice 3 : écrire une fonction qui prend 3 nombres en paramètres et retourne le tuple composé du plus petit des trois et du plus grand des 3 nombres (min, max). Avec 4 exemples d'appels.

Exercice 4 : Supposons que nous ayons une fonction qui prend une liste de nombres en entrée et renvoie à la fois la somme et la moyenne de ces nombres. Nous pouvons utiliser un tuple pour renvoyer deux valeurs.

### Séance 6 : Utiliser des librairies pour s’amuser

1. \*\*Objectifs\*\* : Apprendre à utiliser les dictionnaires et les ensembles pour organiser les données.

2. \*\*Contenu\*\* :

- modifer un tableau excel

- créer un fichier pdf

3. \*\*Exercices\*\* :

- Utiliser ChatGPT pour des questions sur les opérations et les erreurs courantes.

Vous allez utiliser des librairies (des extensions du langage python, qui permet d’aller beaucoup plus loin)

indique moi les modules en python pour manipuler les pdf

indique moi les modules en python pour manipuler les fichier excel

**extension PDF :**

Exercice 1 : Demandez à l’utilisateur de saisir une liste de mots. La saisie se termine quand l’utilisateur saisit « q » pour quitter. Le programme crée un fichier pdf nommé « resultat.pdf » (dans le dossier du programme python) et il affichera sur la première page tous les mots de la liste.

Exercice 2 : M**ê**me programme mais le fichier pdf nommé « resultat2.pdf » aura autant de pages qu’il y a de mot dans la liste, et chaque mot sera afficher au milieu de la page.

Extension Excel :

Exercice 3 : Créez un nouveau fichier excel « resultat.xlsx » qui contient 2 feuilles de calcul. Sur la première feuille, il y a la liste des jours. Sur la deuxième feuille de calcul, de A1 à A10, il y aura les valeurs de 1 à 10, et de B1 à B10, une formule qui double la valeur de la colonne A. le fichier excel sera dans le dossier du programme python.

Exercice 4 : Ouvrir le fichier « resultat.xlsx » précédant et multiplier par 2 tous les nombres de la conne A avant de le sauvegarder.

### Séance 7 : Les dictionnaires et les ensembles

1. \*\*Objectifs\*\* : Apprendre à utiliser les dictionnaires et les ensembles pour organiser les données.

2. \*\*Contenu\*\* :

- Les dictionnaires : création, manipulation, méthodes.

- Les ensembles : création, opérations, méthodes.

3. \*\*Exercices\*\* :

- Travailler avec des dictionnaires pour stocker des paires clé-valeur.

**EXERCICE 3.1**

Je veux gérer un dictionnaire français-anglais

le dictionnaire doit contenir 10 mots.

3.1.1. Votre programme demande des traductions : il propose aléatoirement un mot (français ou anglais) et l’utilisateur répond. Il y a 3 mots à traduire avec une note A, B, C ou D selon le nombre de bonne réponse (de 0 à 3).

3.1.2. Dans un deuxième temps, arrangez-vous pour qu’on ne puisse pas demander 2 fois le même mot à l’utilisateur.

3.1.3. est-il possible d’ajouter l’espagnol pour avoir 3 langues dans le jeu ?

**EXERCICE 3.2**

- Utiliser des ensembles pour des opérations ensemblistes.

Utilisez un fichier excel « liste.xlsx » (nom – prénom – age)

Utilisez 2 fichiers excel « note\_info.xlsx » et « note\_math.xlsx » (nom – prénom – matière – note).

Affichez pour chaque élève son age et sa moyenne générale (moyenne des 2 notes).

**EXERCICE 3.3**

- Traduction automatique de votre programme python en plusieurs langues.

Créez un simple projet python avec 2 fichiers « main.py » et « ma\_lib.py » qui permet dans le main de demander le nom et l’age de l’utilisateur. Et dans votre module « ma\_lib.py » afficher « Bonjour , votre nom est et votre age est : »

* dans votre projet, créer un dossier « traduction »
* installer « Babel »
* dans tous vos fichier « \*.py », importez « from gettext import gettext as \_ »
* et modifiez vos chaines de caractères "bonjour" devient **\_(**"bonjour"**)**
* créer le fichier « babel.cfg »
* chercher les chaines de caractères dans vos fichier « \*.py », et générez le fichier « .pot »
* traduire les textes de votre programmes (grace à « pip install googletrans==4.0.0-rc1 » )
* traduire les fichiers en anglais et espagnol (générer un dossier par langues)

demander à l’IA de vous aider

### Séance 7 : Gestion des fichiers

1. \*\*Objectifs\*\* : Apprendre à lire et écrire des fichiers pour la persistance des données.

2. \*\*Contenu\*\* :

- Ouvrir, lire et écrire des fichiers.

- Les modes d'ouverture des fichiers.

- Gestion des exceptions liées aux fichiers.

3. \*\*Exercices\*\* :

- Lire des données à partir de fichiers texte.

- Écrire des données dans des fichiers.

- Utiliser ChatGPT pour des exemples de gestion de fichiers et résoudre des erreurs.

Exercice : Gestionnaire de journal personnel

Objectif : Créer un programme qui gère un journal personnel stocké dans un fichier texte. Le programme doit permettre à l'utilisateur d'ajouter des entrées, de lire des entrées spécifiques, de rechercher des mots-clés et de générer des statistiques simples.

Voici les fonctionnalités du menu à implémenter :

1. Ajouter une nouvelle entrée au journal
2. Lire une entrée spécifique par date
3. Rechercher des entrées contenant un mot-clé
4. Afficher des statistiques (nombre total d'entrées, dates de la première et de la dernière entrée)
5. Quitter le programme

Chaque entrée du journal doit être au format :[DATE] : [CONTENU]

La gestion des dates est réalisée avec le module datetime.

Votre fichier « journal.txt » sera écrit dans votre dossier « Documents »

Il peut exister (ou pas) au démarrage de votre application

### Séance 8 : Introduction à la programmation orientée objet (POO)

1. \*\*Objectifs\*\* : Comprendre les concepts de base de la POO et leur application en Python.

2. \*\*Contenu\*\* :

- Classes et objets.

- Attributs et méthodes.

- Constructeur (`\_\_init\_\_`).

3. \*\*Exercices\*\* :

- Créer des classes simples et des objets.

- Utiliser ChatGPT pour des questions sur la POO et des exercices de création de classes.

**Exercice 8,1 : Création d'une classe Livre**

Question 1 : Créez une classe Livre avec les attributs suivants :

* titre (chaîne de caractères)
* auteur (chaîne de caractères)
* annee\_publication (entier)
* disponible (booléen, initialisé à True par défaut)

Ajoutez une méthode \_\_init\_\_ pour initialiser ces attributs et une méthode

Exemple d’utilisation du constructeur

livre1 = Livre("1984", "George Orwell", 1949)

livre2 = Livre("Le Petit Prince", "Antoine de Saint-Exupéry", 1943)

afficher\_info() qui affiche toutes les informations du livre.

Question 2 : Ajoutez deux méthodes à la classe Livre :

* emprunter() : change l'état de disponibilité à False et affiche un message approprié.
* rendre() : change l'état de disponibilité à True et affiche un message approprié.

Assurez-vous que ces méthodes vérifient l'état actuel du livre avant de le modifier (par exemple, on ne peut pas emprunter un livre déjà emprunté).

Question 3 : Créez une classe Bibliotheque qui possède :

* Un attribut livres (une liste de Livre)
* Une méthode ajouter\_livre(livre) pour ajouter un livre à la bibliothèque
* Une méthode afficher\_catalogue() qui affiche les informations de tous les livres de la bibliothèque
* Une méthode rechercher\_livre(titre) qui renvoie le livre correspondant au titre donné (ou None s'il n'existe pas)

Testez votre code en créant une bibliothèque, en y ajoutant quelques livres, et en utilisant toutes les méthodes que vous avez créées.

Question 4 : Utilisez la classe « pickle » pour sauver votre bibliothèque lorsque vous ajouter un livre et la recharger lorsque vous démarrer votre programme

### Séance 9 : Projets pratiques

1. \*\*Objectifs\*\* : Appliquer les connaissances acquises à des projets concrets.

2. \*\*Contenu\*\* :

- Présentation de projets simples (jeu, gestion de contacts, etc.).

- Planification et division des tâches.

2. \*\*Exercices\*\* :

- Développer des projets en groupes ou individuellement.

- Utiliser ChatGPT pour obtenir de l'aide sur des parties spécifiques du projet et des questions de débogage.

### Séance 10 : Utilisation d’API IA dans python

1. \*\*Objectifs\*\* : Ecrire un programme python qui utilise directement l’IA

2. \*\*Contenu\*\* :

- Révision des concepts clés.

- Discussion sur des sujets avancés selon les intérêts des étudiants (par exemple, bibliothèques externes, web scraping, etc.).

- Questions et réponses avec ChatGPT.

Exercice 1 : Demandez à l’IA un exemple de programme python qui permet d’interroger l’IA à travers son API.

Exercice 2 : créer un programme en python qui permet à l’utilisateur de sélectionner un fichier python, puis demande à l’IA d’expliquer en 3 lignes ce qu’il fait, et affiche le résultat à l’utilisateur.

### Utilisation de ChatGPT tout au long du cours :

- \*\*Comprendre les erreurs\*\* : Les étudiants peuvent utiliser ChatGPT pour obtenir des explications sur les erreurs qu'ils rencontrent.

- \*\*Exercices et projets\*\* : ChatGPT peut fournir des exemples de code, des conseils pour résoudre des exercices et des suggestions pour améliorer le code.

- \*\*Recherche de documentation\*\* : Les étudiants peuvent utiliser ChatGPT pour rechercher des informations sur des fonctions ou des bibliothèques spécifiques.

- \*\*Aide à la compréhension\*\* : Les étudiants peuvent poser des questions théoriques à ChatGPT pour mieux comprendre les concepts.

En suivant ce plan, vos étudiants devraient acquérir une bonne base en Python et être capables de continuer leur apprentissage de manière autonome, avec l'aide de ChatGPT comme ressource supplémentaire.