* préparé par :

MINYANI MOHAMED-REDA

En-NACHOUA MOURAD

LAGHRYB ADIL

SAHMOUDI ILHAM

* Encadré par :

MS HAQAY

Rapport-POO-G1

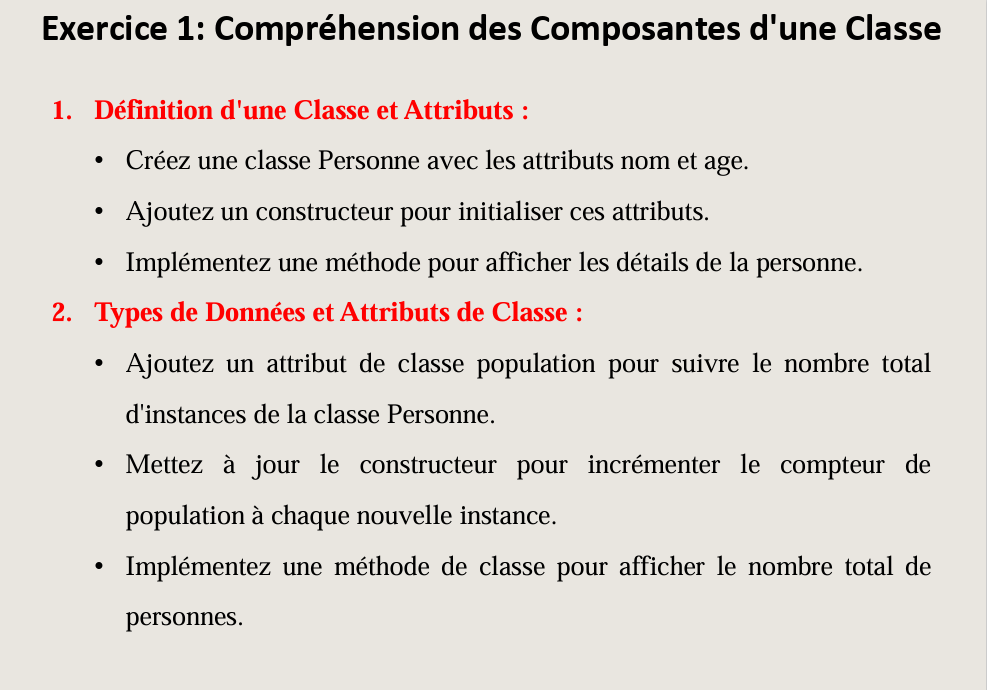
Sommaire :

TP 1

* Exercice 1 : Compréhension des Composantes d'une Classe
* Exercice 2 : Encapsulation et Modificateurs
* Exercice 3 : Définition et Destruction d'un Objet
* Exercice 4 : Méthodes et Visibilité

**TP 2**

* Exercice 1 : Modélisation d'une Classe
* Exercice 2 : Principe de l'Encapsulation
* Exercice 3 : Principe de l'Héritage
* Exercice 4 : Principe du Polymorphisme



Afficher details ()

Afficher nbr ()

Nom : chaine

Âge : entier

Population : entier

personne

La conception

La solution

pData/Local/Programs/Python/Python311/python.exe c:/Users/PC/Desktop/poo/tp1.py

nom:hajar,age:9

nom:aya,age:24

le nombre total de personnes est:2

L’exécution

class personne:

    population=0

    def \_\_init\_\_(self,nom,age):

        self.nom=nom

        self.age=age

        personne.population+=1

    def afficher(self):

        print(f"nom:{self.nom},age:{self.age}")

    @classmethod

    def afficherpopulation(cls):

        print(f"le nombre total de personnes est:{cls.population}")

personne1=personne(nom="sara",age=9)

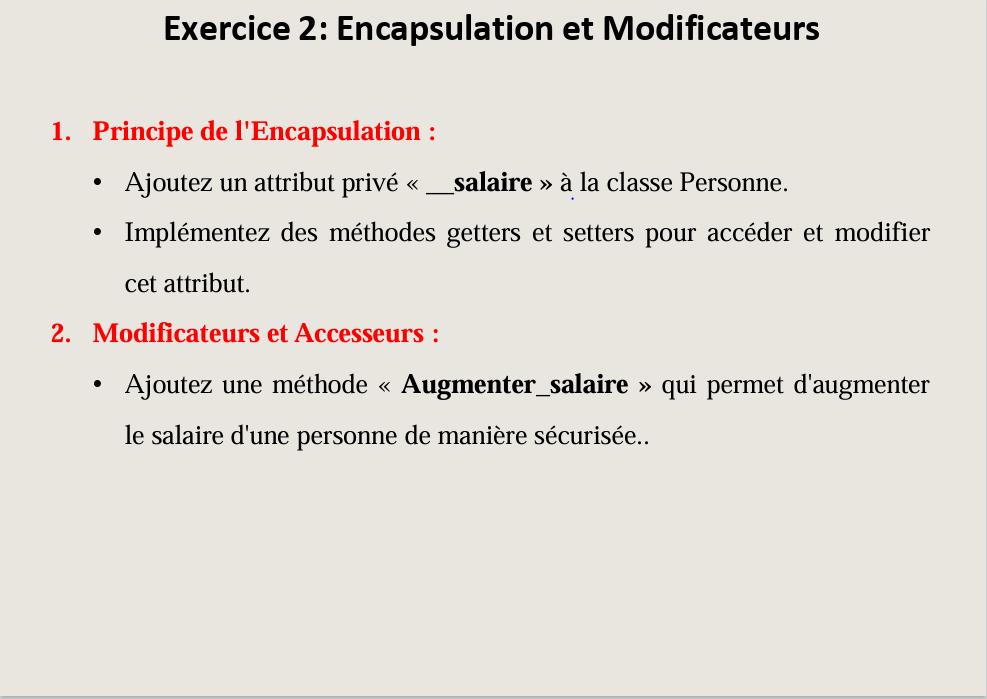
personne2=personne(nom="alae",age=24)

personne1.afficher()

personne2.afficher()

personne.afficherpopulation()

print(personne.population)



Augmenter\_salaire()

Get\_salaire( )

Set\_salaire( )

Nom :chaine

Age :entier

Population :entier

(-)salaire :entier

personne

La conception

La solution

pData/Local/Programs/Python/Python311/python.exe c:/Users/PC/Desktop/poo/main.py

Nom: fatima, Age : 20

Salaire initial: 1550

Nouveau salaire: 1750

L’execution

class Personne:

    population = 0

    def \_\_init\_\_(self, nom, age, salaire):

        self.nom = nom

        self.age = age

        self.\_\_salaire = salaire

        Personne.population += 1

    def afficher(self):

        print(f"Nom: {self.nom}, Age : {self.age}")

    @classmethod

    def afficherpopulation(cls):

        print(f"le nombre total de personnes est: {cls.population}")

    def get\_Salaire(self):

        return self.\_\_salaire

    def set\_Salaire(self, nv):

        self.\_\_salaire = nv

    def Augmenter\_salaire(self, augmente\_argent):

        self.\_\_salaire += augmente\_argent

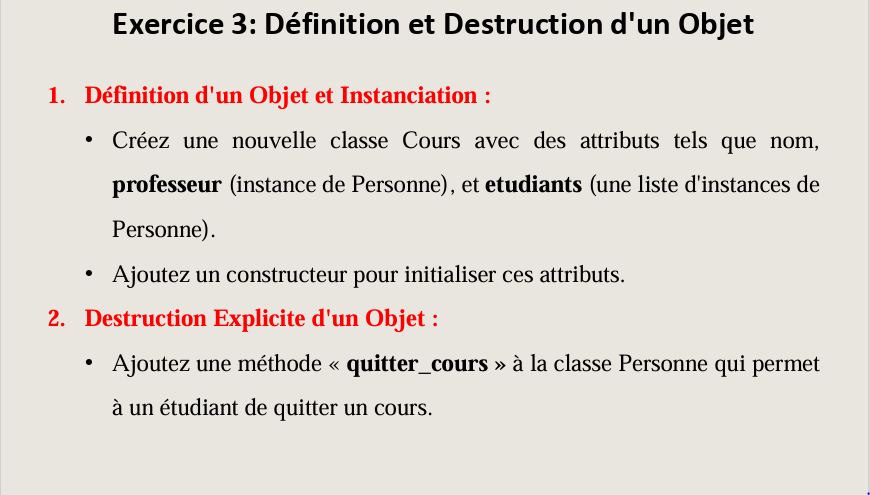
p2=Personne("fatima",20,1550)

p2.afficher()

print("Salaire initial:",p2.get\_Salaire())

p2.Augmenter\_salaire(100)

print("Nouveau salaire:",p2.get\_Salaire())



Quitter\_cours( )

cours

Nom :chaine

Professeur :instance de personne

Etudiants :liste de personne

personne

Nom :chaine

Age :entier

(-)salaire :entier

Population :entier

Get\_salaire( )

Set\_salaire( )

Augmenter\_salaire( )

Quitter\_cours( )

La conception

La solution

empState/0C95E4A579307C6CF93F139E12D62022/SahmoudiIlham.py

salma a quitté le cours chimie.

PS C:\Users\PC\Desktop\poo>

L’exécution

class Cours:

    def \_\_init\_\_(self, nom, professeur):

        self.nom = nom

        self.professeur = professeur

        self.etudiants = []

    def ajouter\_etudiant(self, etudiant):

        self.etudiants.append(etudiant)

    def quitter\_cours(self, etudiant):

        if etudiant in self.etudiants:

            self.etudiants.remove(etudiant)

            print(f"{etudiant.nom} a quitté le cours {self.nom}.")

        else:

            print(f"{etudiant.nom} n'est pas inscrit dans ce cours.")

class Personne:

    def \_\_init\_\_(self, nom):

        self.nom = nom

professeur = Personne("mr madani")

cours = Cours("chimie", professeur)

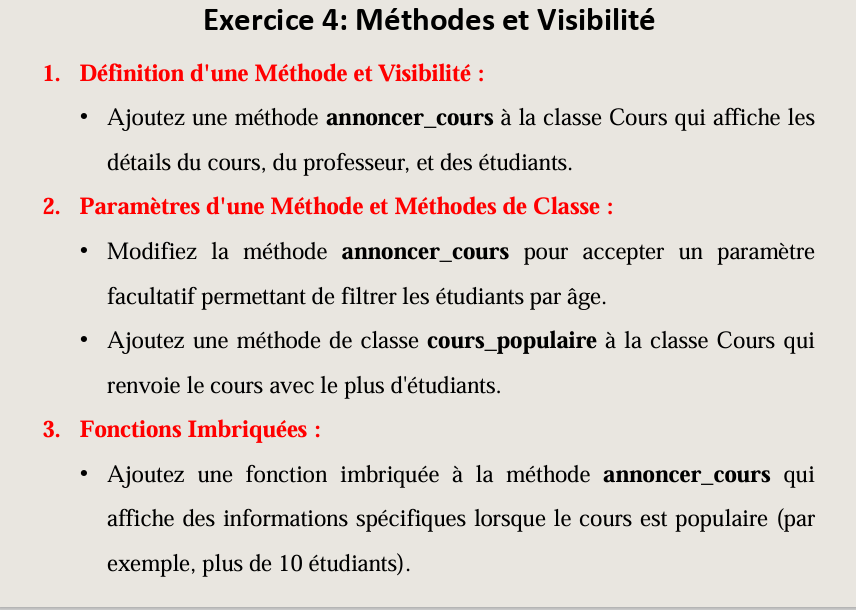
etudiant1 = Personne("salma")

etudiant2 = Personne("sofiane")

cours.ajouter\_etudiant(etudiant1)

cours.ajouter\_etudiant(etudiant2)

cours.quitter\_cours(etudiant1)



Quitter\_cours()

Annoncer\_cours()

Cours\_populaire()

Nom :chaine

Professeur :instance de personne

Etudiants :liste de personne

cours

Get\_salaire( )

Set\_salaire( )

Augmenter\_salaire( )

Nom :chaine

Age :entier

(-)salaire :entier

Population :entier

personne

La solution

class Cours:

    def \_\_init\_\_(self, nom, professeur,etudiants):

        self.nom = nom

        self.professeur = professeur

        self.etudiants = etudiants if not None else[]

    def ajouter\_etudiant(self, etudiant):

        self.etudiants.append(etudiant)

    def annoncer\_cours(self, age=None):

        print(f"Nom du cours : {self.nom}")

        print(f"Professeur : {self.professeur.nom}")

        if age:

            etudiants\_filtres = [etudiant for etudiant in self.etudiants if etudiant.age == age]

            print(f"Étudiants (âge {age}) : {[etudiant.nom for etudiant in etudiants\_filtres]}")

        else:

            print(f"Étudiants : {[etudiant.nom for etudiant in self.etudiants]}")

        if len(self.etudiants) > 10:

            self.\_infos\_cours\_populaire()

    @classmethod

    def cours\_populaire(cls):

        cours\_plus\_populaire = max(cls.cours, key=lambda cours: len(cours.etudiants))

        return cours\_plus\_populaire

    def \_infos\_cours\_populaire(self):

        print("Ce cours est populaire ! Il y a plus de 10 étudiants.")

class Personne:

    def \_\_init\_\_(self, nom, age):

        self.nom = nom

        self.age = age

professeur = Personne("Professeur Salim", 20)

cours1 = Cours("chimie", professeur)

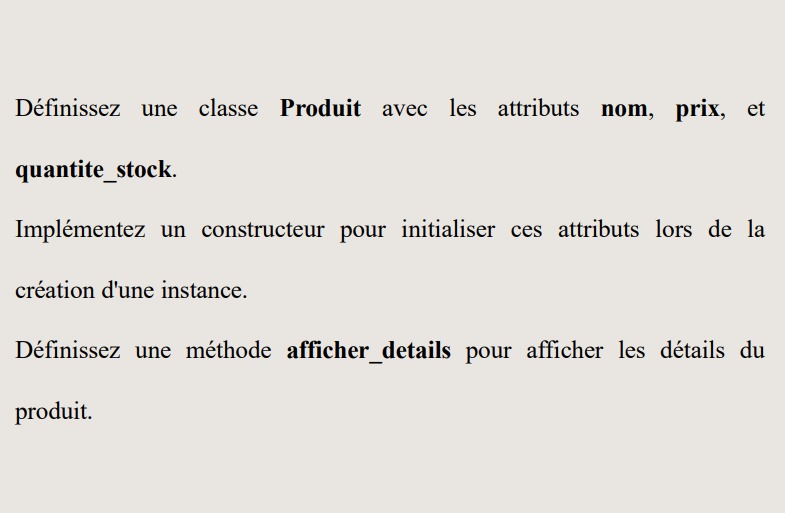
etudiant1 = Personne("Étudiant 1", 18)

etudiant2 = Personne("Étudiant 2", 19)

etudiant3 = Personne("Étudiant 3", 21)

cours1.ajouter\_etudiant(etudiant1)

TP 2



Afficher\_detail( )

Nom :chaine

Prix :chaine

Quantite\_stock :entier

produit

La conception

.exe "c:/Users/PC/Desktop/poo/ex1 tp2

Ex1.py"

nom du produit est:huile,le prix 30 ,la quantite de stock 100

nom du produit est : beurre ,le prix 10 , la quantite de stock 50

L’exécution

class produit:

    def \_\_init\_\_(self,nom=None,prix=0,quantite\_stock=0):

        self.nom=nom

        self.prix=prix

        self.quantite\_stock=quantite\_stock

    def affecher\_detail(self):

        print(f"nom du produit est:{self.nom},le prix {self.prix} ,la quantite de stock {self.quantite\_stock}")

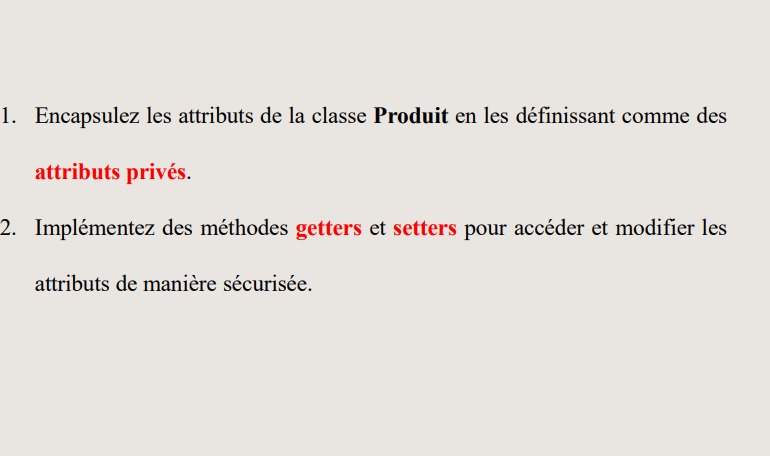
p1=produit("huile",30,100)

p2=produit("beurre",10,50)

p1.affecher\_detail()

p2.affecher\_detail()

La solution



produit

(\_)nom :chaine

(\_)prix :chaine

(\_)quantite\_stock :entier

Afficher\_details( )

Get\_nom( )

Set\_nom( )

Get\_prix( )

Set\_prix( )

Get\_quantite( )

Set\_quantite( )

Sort\_list( )

Print\_star( )

La conception

La solution

class Produit:

    def \_\_init\_\_(self,nom,prix,quantite\_stock):

        self.\_\_nom=nom

        self.\_\_prix=prix

        self.\_\_quantite=quantite\_stock

    def afficher\_details(self):

        return f"Nom est : {self.\_\_nom}\nPrix est :{self.\_\_prix}\nQuantite de stock est : {self.\_\_quantite}"

    def get\_Produit(self):

        return self.\_\_nom

    def get\_Produit(self):

        return self.\_\_prix

    def get\_Produit(self):

        return self.\_\_quantite

    def set\_Produit(self):

        return self.\_\_nom

    def set\_Produit(self):

        return self.\_\_prix

    def set\_Produit(self):

        return self.\_\_quantite

    @staticmethod

    def sort\_list(liste ):

        liste\_decroissante=[]

        while liste:

            liste\_decroissante.append(max(liste))

            liste.remove(max(liste))

            return liste\_decroissante

    @staticmethod

    def print\_Star(n):

        for i in range(n):

            for j in range(i):

                print("\*",end='')

            print('\n')

liste  =[2 , 4 , 5 , 1 , 0]

sort\_list= Produit.sort\_list(liste)

print(sort\_list)

Produit.print\_Star(5)

PS C:\Users\PC\Desktop\poo> & C:/Users/PC/AppData/Local/Programs/Python/Python311/python.exe c:/Users/PC/Desktop/poo/ex3.py

[5]

\*

\*\*

\*\*\*

\*\*\*\*

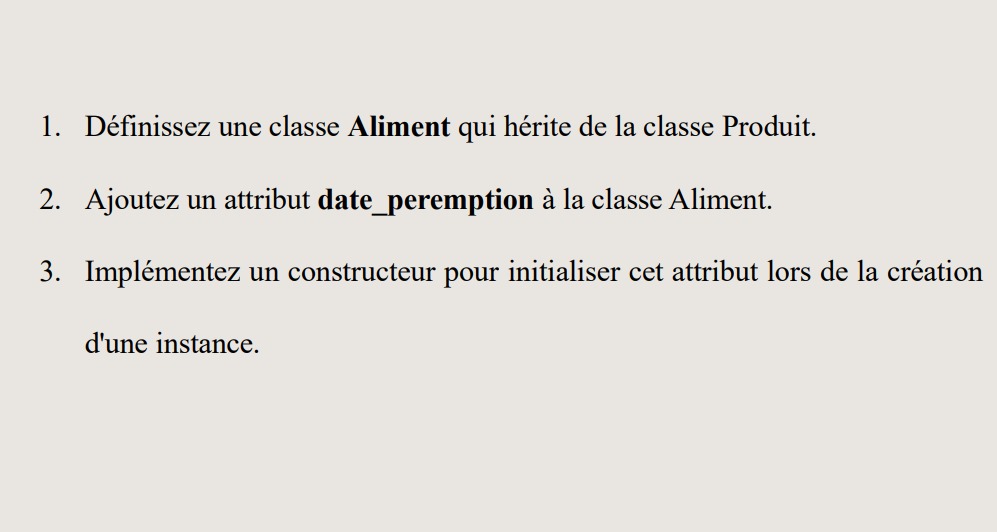
\*

\*\*

\*\*\*

\*\*\*\*

L’exécution



Afficher\_details( )

(#)Nom :chaine

(#)Prix :chaine

(#)quantite\_stock :entier

produit

Afficher\_details( )

Nbr\_objects( )

Afficher\_details( )

Nbr\_objects( )

Date\_peremption :date

aliment

La conception

La solution

class produit:

    def \_\_init\_\_(self,nom=None,prix=0,quantite\_stock=0):

        self.\_nom=nom

        self.\_prix=prix

        self.\_quantite\_stock=quantite\_stock

    def affecher\_detail(self):

        print(f"nom du produit est:{self.\_nom},le prix {self.\_prix} ,la quantite de stock {self.\_quantite\_stock}")

class aliment(produit):

    nbre\_object=0

    def \_\_init\_\_(self,nom,prix,quantite\_stock,date\_peremption):

        super().\_\_init\_\_(nom,prix,quantite\_stock)

        self.date\_peremption=date\_peremption

        aliment.nbre\_object+=1

@classmethod

def calculer\_object(cls):

    print(f"nomber des objects:{cls.nbre\_object}")

p1=produit("sucre",10,177)

p2=produit("farine",18,188)

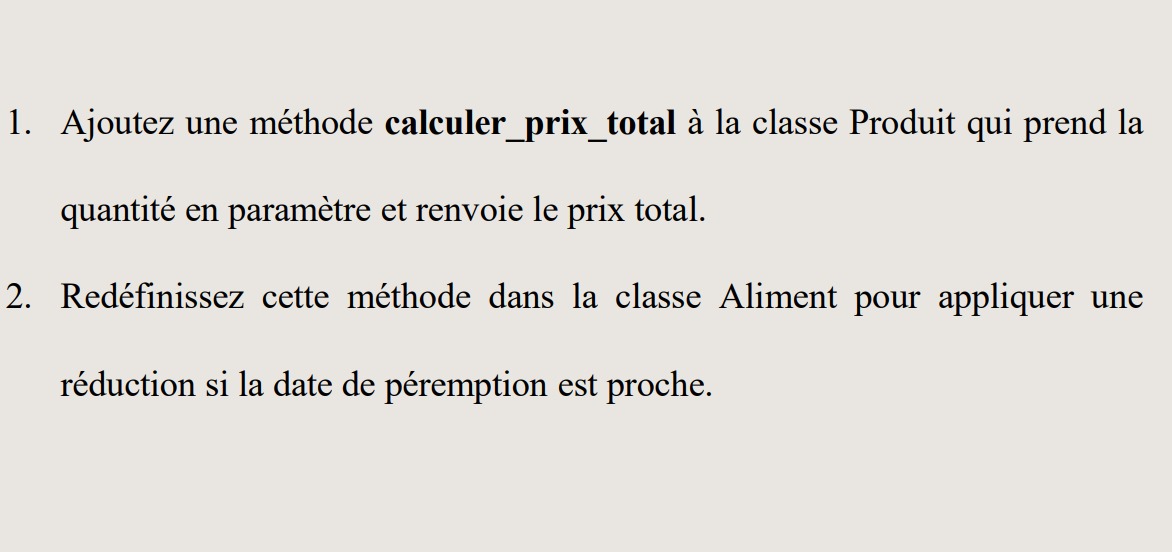
p1.affecher\_detail()

p2.affecher\_detail()

al1=aliment("lait",14,199,"2000-12-04")

print(al1.nbre\_object)

L’execution



\* nom du produit est:sucre,le prix 10 ,la quantite de stock 177

nom du produit est:farine,le prix 18 ,la quantite de stock 188

1

PS C:\Users\PC\Desktop\poo>

\*\*

\*\*\*

\*\*\*\*

\

Calculer\_prix\_total( )

Afficher\_details( )

Nbr\_objects( )

Nom :chaine

Prix :chaine

quantite\_stock :entier

produit

Calculer\_prix\_total( )

Nbr\_objects( )

Afficher\_details( )

Nbr\_objects( )

Date\_peremption :date

aliment

La conception

La solution

from datetime import datetime

class Produit:

    def \_\_init\_\_(self, nom, prix,quantite\_stock):

        self.nom = nom

        self.prix = prix

        self.quantite\_stock=quantite\_stock

    def afficher\_details(self):

        print(f"le nom est:{self.nom},le prix est:{self.prix} ,la quantite de stock:{self.quantite\_stock}")

    def calculer\_prix\_total(self, quantite):

        return self.prix \* quantite

class Aliment(Produit):

    population=0

    def \_\_init\_\_(self, nom, prix,quantite\_stock, date\_peremption):

        super().\_\_init\_\_(nom, prix,quantite\_stock)

        self.date\_peremption = datetime.strptime(date\_peremption, '%Y-%m-%d')

        Aliment.population+=1

    def afficher\_details(self):

        print(f"le nom est :{self.nom},le prix est:{self.prix},la quantite de stock est :{self.quantite\_stock},la date de peremption est:{self.date\_peremption}")

    def calculer\_prix\_total(self, quantite):

        prix\_total = super().calculer\_prix\_total(quantite)

        if self.\_date\_peremption\_proche():

            reduction = 0.1

            prix\_total -= prix\_total \* reduction

        return prix\_total

    def \_date\_peremption\_proche(self):

        seuil\_jours = 7

        delta = self.date\_peremption - datetime.now()

        return delta.days <= seuil\_jours

p1 = Produit("Livre", 10, 10)

print("Prix total du produit:", p1.calculer\_prix\_total(2))

p1.afficher\_details()

al1 = Aliment("Lait", 2, 15, "2024-02-25")

print("Prix total :", al1.calculer\_prix\_total(3))

PS C:\Users\PC\Desktop\poo> & C:/Users/PC/AppData/Local/Programs/Python/Python311/python.exe "c:/Users/PC/Desktop/poo/tes tp.py"

Prix total du produit: 20

le nom est:Livre,le prix est:10 ,la quantite de stock:10

Prix total : 5.4

1

PS C:\Users\PC\Desktop\poo>

\*\*

\*\*\*

\*\*\*\*

L’execution