Correction Examen_LFI_2024

Partie 1

produits = { "Chemise": [29.99, 30], "Pantalon": [39.99, 40], "Jupe": [24.99, 20], "Veste": [59.99, 25],"Robe": [49.99, 35], "Short": [19.99, 15], "Pull": [19.99, 45], "Chaussures": [69.99, 10]}

Partie 2

```
# 1. Ajouter un produit

def AjouterProduit(produits, nom, prix, quantite):
    produits[nom] = [prix, quantite]
    print("Produit ajouté :", nom)


# 2. Rechercher un produit
def RechercherProduit(produits, nom):
    if nom in produits:
        print("Nom :", nom)
        print("Prix :", produits[nom][0])
        print("Quantité :", produits[nom][1])
    else:
        print("Produit non trouvé.")


# 3. Afficher tous les produits
def AfficherStock(produits):
    print("\nListe des produits :")
    for nom in produits:
        print(nom, "-", produits[nom][0], "DT -", produits[nom][1], "en stock")


# 4. Vendre un produit
def VendreProduit(produits, nom, quantite):
    if nom in produits:
```

```
        if produits[nom][1] >= quantite:
            produits[nom][1] -= quantite
            print("Vente effectuée de", quantite, nom)
        else:
            print("Pas assez de stock pour", nom)
    else:
        print("Produit non trouvé.")

# 5. Produit le plus cher
def ProduitLePlusCher(produits):
    max_prix = 0
    produit_cher = ""
    for nom in produits:
        if produits[nom][0] > max_prix:
            max_prix = produits[nom][0]
            produit_cher = nom
    print("Produit le plus cher :", produit_cher, "-", max_prix, "DT")

# 6. Produits en rupture de stock
def ProduitEnRupture(produits):
    print("Produits en rupture de stock :")
    for nom in produits:
        if produits[nom][1] == 0:
            print(nom)

# 7. Rechercher par plage de prix
def RechercherParPrix(produits, prix_min, prix_max):
    print("Produits entre", prix_min, "et", prix_max, "DT :")
    for nom in produits:
```

```
        prix = produits[nom][0]
        if prix_min <= prix <= prix_max:
            print(nom, "-", prix, "DT")

# 8. Appliquer une remise
def AppliquerRemise(produits, nom, pourcentage):
    if nom in produits:
        prix = produits[nom][0]
        nouveau_prix = prix - (prix * pourcentage / 100)
        produits[nom][0] = round(nouveau_prix, 2)
        print("Nouveau prix de", nom, ":", produits[nom][0], "DT")

# 9. Calcul de la valeur totale du stock
def CalculerValeurTotaleStock(produits):
    total = 0
    for nom in produits:
        total += produits[nom][0] * produits[nom][1]
    print("Valeur totale du stock :", round(total, 2), "DT")

# 10. Produits à faible rotation
def IdentifierProduitsBasRotation(produits, seuil):
    print("Produits avec moins de", seuil, "en stock :")
    for nom in produits:
        if produits[nom][1] < seuil:
            print(nom)
def main():
    AfficherStock(produits)
    AjouterProduit(produits, "Gilet", 34.99, 10)
    RechercherProduit(produits, "Veste")
```

```
    VendreProduit(produits, "Chemise", 5)
    AppliquerRemise(produits, "Robe", 10)
    ProduitLePlusCher(produits)
    ProduitEnRupture(produits)
    RechercherParPrix(produits, 20, 50)
    CalculerValeurTotaleStock(produits)
    IdentifierProduitsBasRotation(produits, 20)

# Lancer le programme
main()
```

Exercice 2

Exercice 2 : Le problème du sac à dos (8 points)

Étant donné les poids et les valeurs de **n** articles, nous devons mettre ces articles dans un sac à dos de capacité **C** pour obtenir la valeur totale maximale dans le sac à dos.

Une solution efficace consiste à utiliser l'approche gloutonne. L'idée de l'approche gloutonne est de calculer le rapport **valeur/poids** pour chaque article et de trier les articles sur la base de ce rapport. Ensuite, prenez l'élément avec le rapport le plus élevé et ajoutez-le jusqu'à ce que nous ne puissions plus ajouter l'élément suivant dans son intégralité et à la fin, ajoutez-le autant que possible.

Exemple:

```
articles = [(60, 10), (100, 20), (100, 60), (40, 20)]
capacite = 50
```

Le script affichera:

```
Sac_a_dos = [(60, 10), (100, 20), (40, 20)]
La valeur maximale du sac à dos = 200.0
```

Travail demandé:

Ecrire un script Python qui permet de résoudre le problème selon l'approche proposée.

```
# Fonction pour obtenir le rapport valeur/poids
def obtenir_rapport(article):
    valeur = article[0]
    poids = article[1]
    return valeur / poids
```

```
# Fonction principale pour remplir le sac
def sac_a_dos(articles, capacite):
    # Liste pour le contenu du sac
    sac = []

    # Trier les articles par rapport décroissant (du plus rentable au moins rentable)
    articles.sort(key=obtenir_rapport, reverse=True)

    valeur_totale = 0.0

for article in articles:
    valeur = article[0]
    poids = article[1]

    if capacite >= poids:
        sac.append((valeur, poids))
        capacite -= poids
        valeur_totale += valeur
    else:
        fraction = capacite / poids
        valeur_partielle = valeur * fraction
        sac.append((valeur_partielle, capacite))
        valeur_totale += valeur_partielle
        break

    print("Sac_a_dos =", sac)
    print("La valeur maximale du sac à dos =", round(valeur_totale, 2))

# Liste des articles : [valeur, poids]
```

```
articles = [
    [60, 10],
    [100, 20],
    [100, 60],
    [40, 20]
]

capacite = 50

# Appel de la fonction
sac_a_dos(articles, capacite)
```