DS 1 : lundi 3 novembre

2h sans calculatrice, le sujet est composé de deux parties indépendantes.

N.B. : le candidat attachera la plus grande importance à la clarté, à la précision et à la concision de la rédaction. Si un candidat est amené à repérer ce qui peut lui sembler être une erreur d'énoncé, il le signalera sur sa copie et devra poursuivre sa composition en expliquant les raisons des initiatives qu'il a été amené à prendre.

Optimisation de rendement d'une entreprise de livraison

Une entreprise de livraison dispose de plusieurs locaux en France et chacun possède plusieurs camions de livraisons. Celle-ci souhaite optimiser le chargement de ses camions pour diminuer ses frais de fonctionnement.

Partie I - Données liées aux livraisons conservées par l'entreprise

À chaque livraison, l'entreprise stocke des données relatives à celle-ci. L'entreprise dispose de 20 locaux, numérotés de 1 à 20 , disposant chacun d'un certain nombre de camions. Pour faciliter ses livraisons, l'entreprise découpe la France en 30 zones et associe à chaque local, trois zones possibles de livraisons.

Ces données sont enregistrées dans une base de données contenant trois tables :

La table livraison constituée des champs suivants :

— id : identifiant de livraison (entier) ;
— date : date de la livraison au format "jj-mm-aaaa" (chaine de caractères) ;
— heure : heure de la livraison au format : "hh-mm-ss" (chaine de caractères) ;
— id_client : identifiant du client recevant la livraison (entier) ;
— id_local : identifiant du local de l'entreprise (entier compris entre 1 et 20).

La table client constituée des champs suivants :

— id : identifiant du client (entier) ;
— zone : entier compris entre 1 et 30.

La table local constituée des champs suivants :

— id : identifiant du local (entier compris entre 1 et 20) ;
— zonel : entier ;
— zone2 : entier ;
— zone3 : entier.

Q1. Donner une clé primaire pour la table livraison. Donner les clés étrangères de la table livraison.

Q2. Écrire une requête SQL permettant d'obtenir les identifiants des clients livrés le 10 janvier 2021.

Q3. Lister les identifiants des locaux, *par ordre croissant*, qui peuvent livrer dans la zone 10.

Q4. Écrire une requête SQL permettant de récupérer les dates et les heures de toutes les livraisons ayant eu lieu dans la zone 5 le 2 mars 2021.

Q5. Trouver les identifiants des clients livrés par un local dont la zone1 est la zone 15. On fera apparaître chaque identifiant de manière unique (pas de répétition des identifiants)

Q6. Écrire une requête SQL permettant de compter le nombre de livraisons effectuées le 3 février 2021 par des camions dont les locaux ne livrent que dans des zones possibles inférieures ou égales à dix.

Q7. Quels sont les clients ayant effectué au moins 5 livraisons en 2021, et combien de livraisons chacun a-t-il réalisées cette année-là ?

Q8. Lister les identifiants des locaux qui n'ont effectué aucune livraison dans leur zone1 pendant le premier trimestre 2021 .

Q9. Trouver les identifiants des livraisons réalisées le 16 juin 2020 pour lesquelles la zone du client livré est supérieure à la moyenne des zones des clients livrés ce jour là

Partie II - Optimisation du chargement

Chaque camion de l'entreprise peut charger une cargaison jusqu'à un poids maximal noté P_{max} . L'entreprise dispose de différentes informations provenant de ses clients :

— le poids de chaque produit p_i (chaque client propose un seul produit) ;
— la valeur v_i associée au transport de chaque produit : c'est-à-dire l'argent gagné par l'entreprise si elle réalise le transport de ce produit.

En considérant que l'entreprise dispose de n clients, l'entreprise cherche donc à trouver une liste d'indices notée I contenue dans $\{1, \dots, n\}$ telle que :

$$\sum_{i \in I} p_i \leq \mathrm{P}_{\max} \quad \text{(respect du poids maximal)} \tag{1}$$

et

$$\sum_{i \in I} v_i \text{ soit maximal (optimation du profit pour l'entreprise).} \tag{2}$$

Dans toute la suite, les poids seront donnés en centaines de kilogrammes et les valeurs en centaines d'euros.

II. 1 - Un exemple

Dans cette sous-partie, on suppose que $n = 4$ et que $\mathrm{P}_{\max} = 8$ centaines de kilogrammes. On stocke alors les différentes informations dans trois listes :

— Pr est la liste des produits proposés par les clients numérotés de 1 à 4 : $\mathrm{Pr} = [1, 2, 3, 4]$;
— P est la liste des poids associés : $\mathrm{P} = [3, 2, 1, 4]$;
— V est la liste des valeurs associées : $\mathrm{V} = [4, 3, 1, 9]$.

Par exemple, le produit 2 a un poids de deux centaines de kilogrammes et une valeur de trois centaines d'euros.

Les questions **Q10**, **Q11** et **Q12** se traitent à l'aide de calculs simples, à faire à la main.

Q10. Expliquer pourquoi une cargaison constituée d'un, de deux ou de quatre produits ne répond pas au problème posé, c'est-à-dire ne maximise pas le profit fait par l'entreprise en respectant la condition donnée sur le poids maximal.

Q11. Donner toutes les cargaisons de trois produits respectant le poids maximal. On donnera à chaque fois le profit fait par l'entreprise.

Q12. Quelle est la cargaison maximisant le profit de l'entreprise ? Que vaut le profit dans ce cas ?

II. 2 - Une méthode intuitive pour la résolution du problème

On garde les notations de la sous-partie précédente dans le cas général :

— Pr est la liste des produits (numérotés de 1 à n inclus) ;
— $\mathrm{P} = [p_1, \dots, p_n]$ est la liste des poids associés aux produits ;
— $\mathrm{V} = [v_1, \dots, v_n]$ est la liste des valeurs associées aux produits.

Q13. Définir une fonction ListeProduits ayant pour argument un entier naturel non nul n et renvoyant la liste Pr.

Une méthode intuitive pour tenter d'optimiser le profit de l'entreprise est la suivante : on calcule les ratios $\frac{v_i}{p_i}$, puis on trie les objets par ordre décroissant suivant ces valeurs. Les produits sont alors classés par rentabilité : le premier produit devient le plus rentable " au poids " et ainsi de suite. On ajoute progressivement chaque produit dans la cargaison, dans cet ordre, sans dépasser la limite du poids maximal.

Q14. Définir une fonction Ratio ayant pour arguments deux listes P, V où P correspond à la liste des poids et V correspond à la liste des valeurs, renvoyant la liste des ratios $\frac{v_i}{p_i}$.

La fonction suivante est associée à une méthode de tri :

```
def Tri(L):
''' L est une liste de nombres réels '''
    for i in range(1,len(L)):
        x=L[i]
        j=i
        while j>0 and x<L[j-1]:
            L[j]=L[j-1]
            j=j-1
        L[j]=x
    return L
```

Q15. On exécute Tri(L) avec L = $[3, 5, 2, 1]$. Combien y a t-il d'itérations de la boucle for ? Donner la valeur de L à la fin de chaque itération de la boucle for.

Q16. Ce tri fonctionnerait-t-il pour une chaîne de caractères dont chaque caractère est un entier ? Justifier.

Q17. Quelle est la méthode de tri utilisée dans la fonction Tri ? Donner sa complexité (en nombre de comparaisons) dans le pire des cas et dans le meilleur des cas. *On justifiera soigneusement les complexités données.*

Q18. Définir une fonction Inverse ayant pour argument une liste de nombres réels L et renvoyant l'inverse de celle-ci. Par exemple, l'inverse de $[1, 5, 3, 4]$ est $[4, 3, 5, 1]$.

L'utilisation de $L[::-1]$ n'est pas autorisée.

Q19. On souhaite trier une liste de poids P et une liste de valeurs V associées à une liste de produits en suivant l'ordre décroissant de la liste des ratios $\frac{v_i}{p_i}$. Justifier que les fonctions Ratio, Tri et Inverse ne permettent pas de répondre simplement au problème posé.

Q20. Écrire, à l'aide des fonctions Ratio et Inverse, une fonction Tri2 ayant pour arguments une liste de poids P et une liste de valeurs V associées à une liste de produits. Cette fonction renverra les listes de poids et de valeurs triées par ordre décroissant de la liste des ratios.

Q21. Compléter (inutile de recopier tous le code, donner juste (A) et (B)) la définition de la fonction Vmax ayant pour arguments les listes de poids P et de valeurs V et le poids maximal $\mathrm{P_{max}}$ du chargement et renvoyant la valeur maximale du profit de l'entreprise en suivant la méthode proposée.

```
def Vmax(P,V,Pmax) :
    P2,V2=Tri2(P,V)
    SP=0
    SV=0
    i=0
    while ..............................................(A).
        SP=SP+P2[i]
        SV=SV+V2[i]
        i=i+1
    return .............................................(B).
```

Q22. On souhaite appliquer cette méthode en utilisant les listes de poids et de valeurs de la souspartie I.1. Donner la liste des ratios, les listes de poids et de valeurs obtenues à l'aide de la fonction Tri2 ainsi que le profit obtenu. Commenter le résultat.

II. 3 - Une méthode récursive

Nous gardons les notations du cas général de la sous-partie I. 2 : Pr, P et V. On considère pour simplifier que les poids des produits sont des entiers, ainsi que $\mathrm{P_{max}}$.

Nous introduisons une méthode récursive pour résoudre le problème d'optimisation :

— pour chacun des produits, deux choix sont possibles : il fait partie de la cargaison ou non ;

— la récursivité s'effectuera sur la liste des indices de Pr : le premier appel de la fonction se fera en utilisant l'indice n, puis l'indice $n-1$ et ainsi de suite jusqu'à l'indice 0 (correspondant au cas où il n'y a plus de produits) ;

— pour $i \in \{0, 1, \ldots, n\}$ et $\omega \in \{0, 1, \ldots, \mathrm{P_{max}}\}$, on note $S(i, \omega)$ la valeur maximale cumulée des produits que l'on peut placer dans un camion d'une capacité maximale (en poids) de ω avec la liste constituée des i premiers produits de Pr.

On pose alors la relation de récursivité suivante :

$$S(i,\omega) = \begin{cases} 0 & \text{si } i = 0 \\ S(i-1,\omega) & \text{si } i > 0 \text{ et } p_i > \omega \\ \max\left(S(i-1,\omega), v_i + S\left(i-1,\omega-p_i\right)\right) & \text{si } i > 0 \text{ et } p_i \leq \omega \end{cases} \tag{3}$$

Q23. Justifier les relations précédentes dans les trois cas.

Q24. Justifier la terminaison de l'algorithme associé à la relation de récursivité précédente, sachant que la première valeur donnée pour i sera n et la première valeur pour ω sera Pmax.

Q25. Définir une fonction Max ayant pour arguments deux réels et renvoyant le maximum parmi ces deux valeurs. Il est interdit d'utiliser la fonction max prédéfinie dans Python.

Q26. En vous basant sur la relation (3), compléter (inutile de recopier tous le code, donner juste (C) et (D)) la définition de la fonction récursive recur ayant pour arguments les listes de poids et de valeurs P et V , un indice i, un poids ω, et renvoyant $S(i,\omega)$.

```
def recur(P,V,i,w) :
    if i==0:
        return .....(C).
    if P[i-1]>w:
        return recur(P,V,i-1,w)
    else:
        .....(D).
```

Q27. Donner une série d'instructions utilisant la fonction recur et permettant de déterminer le profit de la sous-partie I.1.

II. 4 - Amélioration de la méthode récursive

On souhaite améliorer la méthode récursive de la sous-partie I.3. Nous allons procéder en mémorisant des calculs déjà effectués. Voici le principe : nous allons stocker les valeurs $S(i,\omega)$ dans un tableau Memoire, de taille $(n+1) \times (\mathrm{P}_{\max}+1)$, initialisé au départ avec des éléments tous égaux à -1 .

Si la valeur de $S(i,\omega)$ a déjà été calculée, l'élément d'indice (i,ω) de ce tableau Memoire ne sera plus égal à -1 : on renverra donc directement la valeur. Sinon, on la calculera en suivant le principe de la sous-partie I. 3 et on la stockera dans le tableau avant de la renvoyer.

Nous faisons le choix de représenter les tableaux comme des listes de listes.

Q28. Donner l'instruction permettant de créer le tableau Memoire initialisé avec des coefficients égaux à -1 , en supposant Pmax et n connus.

Q29. Compléter (inutile de recopier tous le code, donner juste (E), (F), (G) et (H)) la fonction recur2 en suivant le principe expliqué et permettant d'améliorer la fonction recur. La variable Memoire sera utilisée comme une variable globale.

```
def recur2(P,V,i,w,Memoire) :
    if i==0:
        return 0
    if Memoire[i][w]>-1:
        return .....(E).
    if P[i-1]>w:
        Memoire[i][w]=recur2(P,V,i-1,w,Memoire)
        return Memoire[i][w]
    else:
        if Memoire[i-1][w]==-1:
            Memoire[i-1][w]= .....(F).
        if .....(G).
            Memoire[i-1][w-P[i-1]]=recur2(P,V,i-1,w-P[i-1],Memoire)
            a=max(Memoire[i-1][w],V[i-1]+Memoire[i-1][w-P[i-1]])
            Memoire[i][w]=a
    return .....(H).
```

Avec les données suivantes :
— $\mathrm{P} = [5,3,3,3]$;
— $\mathrm{V} = [4,3,1,1]$;
— $\mathrm{P}_{\max} = 8$;
— Memoire un tableau de taille 5×9 ;
et en exécutant recur2 $(P, V, \mathrm{len}(P), \mathrm{Pmax}$, Memoire), on obtient alors la valeur 7.