Exercice **Gestion de stocks avec des dictionnaires.**

On travaille sur des stocks ou des commandes de produits . Ces stocks et ces commandes sont gérés en utilisant des dictionnaires. Copier et coller ces dictionnaires qui serviront à tester les fonctions.

```
stock = {"crayon":43,"stylobleu":23,"stylorouge":40,"gomme":12,"regle":25, "compas":10}

commande1 = {"crayon": 3,"stylovert":23,"gomme":22,"regle":5}

commande2 = {"crayon" : 2, "feutre" : 1}

prix = {'regle': 5, 'compas': 6, 'gomme': 1.2, 'crayon': 1.1, 'stylorouge': 3 ,
'stylobleu': 2.5, 'stylovert' : 4, 'feutre' : 5}
```

1. Écrire une fonction **nb_objets(commande)** qui prend comme argument une commande et renvoie le nombre total d'objets de cette commande. Pour tester on vérifiera que :
   `nb_objets (commande1)` renvoie 53 ;
   `nb_objets (commande2)` renvoie 3 .

2. On dispose d'un dictionnaire de prix. Ce dictionnaire est composé de clés qui sont des objets (représentés par des strings) et de valeur qui sont des float. Dans l'exemple donné, on voit que la prix correspondant à une règle est 5 euros, et ainsi de suite.

   (a) Écrire une fonction **augmente(dico)** qui prend comme argument un dictionnaire de prix, et augmente tous les prix de 10 %. Dans la console, saisir :
   `>>> augmente(prix)`
   `>>> prix`
   On doit alors voir afficher approximativement :

   ```
   {'regle': 5.5, 'compas': 6.60, 'gomme': 1.32,
   'crayon': 1.21, 'stylorouge': 3.30, 'stylobleu': 2.75}
   ```

   Remarque : on peut améliorer cette fonction en passant en argument le taux d'augmentation en pourcentage. L'appel de la fonction se ferait alors avec `augmente(prix,10)`. Réaliser cette modification et faire des tests.

   (b) Écrire une fonction **facture(com, prix)** qui à partir d'une commande et de prix sous forme de dictionnaires renvoie le montant total de la facture (on suppose que les prix de tous les objets sont dans le dictionnaire prix). Effectuer les tests suivants :
   `facture(commande1,prix)` renvoie 146.7 ;
   `facture(commande2,prix)` renvoie 7.2 .

3. Écrire une fonction **maxi(com)** qui prend comme argument un dictionnaire de commandes, et renvoie le nom d'un des objets dont on a le plus. On renverra un seul nom (n'importe lequel si plusieurs objets réalisent ce maximum). Effectuer les tests suivants à la console :
   `maxi(commande1)` renvoie **stylovert** ;
   `maxi(commande2)` renvoie **crayon** .

4. Un client passe une commande. Écrire une fonction **ote(stock,com)** qui met à jour le stock. La fonction devra écrire un message si le stock est insuffisant ou si le produit n'existe pas.
   Par exemple `>>> ote(stock,commande1)` renvoie :

```
Pas de stylovert dans le stock
Pas assez de gomme , il en manque : 10
```

   et `stock` doit devenir :

```
{'crayon': 40, 'stylobleu': 23, 'stylorouge': 40, 'gomme': 0, 'regle': 20, 'compas': 10}
```

   Faire aussi le test avec `>>> ote(stock,commande2)`

5. Écrire une fonction **add_commande(com1, com2)** qui additionne deux commandes . On renvoie une commande qui regroupe les deux commandes : tous les objets doivent être présents et dans la quantité globale.
   Par exemple, si on saisit dans la console :
   `>>>add_commande(commande1, commande2)` on doit obtenir :

```
{'crayon': 5, 'stylovert': 23, 'gomme': 22, 'regle': 5, 'feutre': 1}
```

   Attention : il ne faut pas que les commandes passées en argument soient changées!!!

   Remarque : la fonction de cet exercice est un exemple de **fusion** de deux dictionnaires. La méthode "update" n'est pas adaptée, de même que la commande "+", qui fonctionne pour les listes mais pas pour les dictionnaires.