Exercices - Modularité

Pour chacun des 3 exercices de ce TD, vous devrez, sur papier, :

- Ecrire un module python
- Ecrire un programme principal permettant de tester votre module.
- ne pas oublier la docstring des fonctions du module python.

Exercice 1 Opérations mathématiques

1. Créez un module nommé `operations_mathematiques.py` en implémentant les fonctions suivantes :
   - Une fonction `addition(a, b)` qui prend deux nombres en entrée et renvoie leur somme.
   - Une fonction `soustraction(a, b)` qui prend deux nombres en entrée et renvoie leur différence.
   - Une fonction `multiplication(a, b)` qui prend deux nombres en entrée et renvoie leur produit.
   - Une fonction
   - division(a, b) qui prend deux nombres en entrée et renvoie leur quotient.
   - Une fonction `puissance(a, b)` qui prend deux nombres en entrée et renvoie a élevé à la puissance b.
2. Créez un programme Python (par exemple, `test_operations_mathematiques.py`) dans lequel vous importez le module `operations_mathematiques` et testez les fonctions avec différents jeux de nombres. Assurez-vous de tester les cas d'addition, de soustraction, de multiplication, de division, et d'élévation à la puissance.

Exercice 2 Probabilités

1. Créez un module nommé `probabilites.py` avec les spécifications suivantes :
   - Une fonction `probabilite_evenement(issues_favorables, issues_totales)` qui prend en entrée le nombre d'issues favorables de l'événement et le nombre total d'issue de cette expérience aléatoire que l'on considère équiprobable, et renvoie la probabilité de l'événement.
   - On suppose que les événements A et B ne sont pas indépendants.
     Une fonction `probabilite_A_inter_B(P_A, P_B_sachant_A)` qui prend en entrée les probabilités des événements A et $B|A$, et renvoie la probabilité de $p(A \cap B)$.
     On donne $P(B|A) = \dfrac{P(A \cap B)}{P(A)}$
   - Une fonction `probabilite_A_union_B(P_A,P_B_sachant_non_A)` qui prend en entrée les probabilités des événements A et $B|\bar{A}$, et renvoie la probabilité de $p(A \cup B)$.
     On donne : $P(A \cup B) = P(A) + P(B) - P(A \cap B)$ et $P(B) = P(A)P(B|A) + P(\bar{A})P(B|\bar{A})$
2. Application : Soit deux programmes informatiques x et y. On notera les événements suivants :
   - X : le programme x fonctionne.
   - $\bar{X}$: le programme x ne fonctionne pas.
   - Y : le programme y fonctionne.
   - $\bar{Y}$: le programme y ne fonctionne pas.

Quelques données supplémentaires :

- La probabilité que le programme x tombe en panne est de 0,3.
- La probabilité que le programme y tombe en panne sachant que le programme x n'est pas tombé en panne est de 0,45.
- La probabilité que le programme y ne tombe pas en panne sachant que le programme x est tombé en panne est de 0,28.

(a) Représenter cette situation par un arbre de probabilité.

(b) Ecrire un programme principal en python qui calcule :

- la probabilité que les deux programmes ne tombent pas en panne. Quel est le résultat attendu ?
- la probabilité que le programme x est en panne et le programme y n'est pas en panne. Quel est le résultat attendu ?
- la probabilité que le programme x est fonctionne ou que le programme y fonctionne. Quel est le résultat attendu ?
- la probabilité que le programme y fonctionne. Quel est le résultat attendu ?

Exercice 3

1. Créez un module nommé `gestion_taches.py` avec les spécifications suivantes :
   - Une fonction `ajouter_tache(tache, liste_taches)` qui prend en entrée une tâche et une liste de tâches, et renvoie la liste mise à jour avec la nouvelle tâche ajoutée.
   - Une fonction `supprimer_tache(tache, liste_taches)` qui prend en entrée une tâche et une liste de tâches, et renvoie la liste mise à jour avec la tâche supprimée.
   - Une fonction `afficher_taches(liste_taches)` qui prend en entrée une liste de tâches, et affiche la liste. Le nombre de tache devra être également affiché.
2. Programme de test : Créez un script Python (par exemple, `test_gestion_taches.py`) dans lequel vous importez le module `gestion_taches` et testez les fonctions avec différentes tâches et listes de tâches. Assurez-vous de tester les cas d'ajout et de suppression.