Amérique du Nord – 2026 – sujet1 - Correction

Exercice 3 (8 points)

Question 1

Le numéro d'un immeuble dans une rue ne peut pas être choisi comme clé primaire car plusieurs rues différentes peuvent contenir un immeuble portant le même numéro.

Par exemple :

- 12 rue de la mer
- 12 rue Turing

existent simultanément.

👉*Commentaire : une clé primaire doit identifier de manière unique chaque enregistrement de la table.*

Question 2

```sql
SELECT id_immeuble
FROM immeuble
WHERE rue_immeuble = 'la mer'
ORDER BY id_immeuble;
```

👉*Commentaire : ORDER BY permet de trier les résultats dans l'ordre croissant par défaut.*

Question 3

```sql
SELECT id_appart
FROM appartement
WHERE id_immeuble = 16
AND etage_appart >= 5;
```

👉*Commentaire : AND permet de combiner plusieurs conditions dans la clause WHERE.*

Question 4

La requête risque de rompre l'intégrité de la base de données car certains appartements peuvent encore référencer l'immeuble 16 grâce à la clé étrangère `id_immeuble`.

Si l'immeuble est supprimé alors que des appartements y sont encore liés, certaines références deviendraient invalides.

👉*Commentaire : une clé étrangère doit toujours référencer un enregistrement existant.*

Question 5

```sql
INSERT INTO immeuble
VALUES (140, 6, 13, 'Turing');
```

👉*Commentaire : INSERT INTO permet d'ajouter un nouvel enregistrement dans une table.*

Question 6

```sql
UPDATE appartement
SET prix_appart = prix_appart * 2
WHERE id_appart = 603;
```

👉*Commentaire : UPDATE modifie des valeurs existantes dans la table. Ne pas oublier la clause SET.*

Question 7

```sql
SELECT MAX(prix_appart)
FROM appartement
JOIN immeuble
ON appartement.id_immeuble = immeuble.id_immeuble
WHERE rue_immeuble = 'la mer';
```

👉*Commentaires :*

- La jointure permet d'associer chaque appartement à son immeuble.
- La fonction d'agrégation MAX permet d'obtenir la plus grande valeur d'un attribut, elle était donnée dans le sujet (pas à connaître em terminale).

PARTIE B — Sous-séquences croissantes

Question 8

Liste :

`L2 = [3, 1, 8, 2, 5]`

Les sous-séquences strictement croissantes de longueur 2 sont :

- [3, 8]
- [3, 5]
- [1, 8]
- [1, 2]
- [1, 5]
- [2, 5]

👉*Commentaire : il faut conserver l'ordre des éléments de la liste initiale.*

Question 9

Une plus longue sous-séquence strictement croissante est : `[1, 2, 5]`

Sa longueur vaut 3.

👉*Commentaire : aucune sous-séquence strictement croissante de longueur 4 n'existe dans cette liste.*

Question 10

```
def est_strict_croissante(seq):
    for i in range(len(seq)-1):
        if seq[i] >= seq[i+1]:
            return False
    return True
```

👉*Commentaires :*

- Chaque élément doit être strictement inférieur au suivant.
- Dès qu'une condition n'est pas respectée, la fonction peut renvoyer False immédiatement.

Question 11

```
def llsc_fin(i):
    if i == 0:
        return 1
    maxi = 1
    for j in range(i):
        if tab[j] < tab[i]:
            maxi = max(maxi, 1 + llsc_fin(j))
    return maxi
```

👉*Commentaire :*

- *On cherche la meilleure sous-séquence se terminant avant l'indice i.*
- L'appel *récursif permet de résoudre le problème sur des sous-problèmes plus petits (sous problème).*

Question 12

```
def llsc_dyn(tab):
    dyn = [1 for i in range(len(tab))]
    for i in range(len(tab)):
        for j in range(i):
            if tab[j] < tab[i]:
                dyn[i] = max(dyn[i], dyn[j] + 1)
    return max(dyn)
```

👉*Commentaire :*

- `dyn[i]` *contient la longueur de la meilleure sous-séquence se terminant à l'indice i.*
- La *programmation dynamique évite de recalculer plusieurs fois les mêmes résultats.*

Question 13

L'implémentation dynamique est plus efficace car elle évite les nombreux calculs répétés effectués par la version récursive.

Elle nécessite donc moins de temps d'exécution.

👉*Commentaire : la programmation dynamique mémorise les résultats intermédiaires afin de les réutiliser.*
