Centres étrangers – 2025 – sujet1 - Correction

Exercice 1 (6 points)

Question 1

```
balise12 = Balise(12, ['vert', 'noir'])
```

👉*Commentaire : D'après le graphe, la balise 12 appartient à l'itinéraire vert et noir.*

Question 2

```
balise9.voisines = [balise8, balise11, balise12]
```

👉*Commentaire : Selon le graphe, la balise 9 est reliée aux balises 8, 11 et 12.*

Question 3

Réponse: `[2, 5, 6]`

Question 4

```
['noir', 'vert']
```

👉*Commentaire : La première méthode ajoute 'noir' si non présent, la deuxième ajoute 'vert'.*

Question 5

```
def itineraire(balise_debut, balise_fin, couleur):
     assert couleur in balise_debut.couleurs_balise
     assert couleur in balise_fin.couleurs_balise
     balise = balise_debut
     chemin = [balise]
     while balise.num_balise != balise_fin.num_balise:
         for b in balise.voisines:
             if (couleur in b.couleurs_balise) and (b not in chemin):
                 balise = b
                 chemin.append(balise)
     return [b.num_balise for b in chemin]
```

👉*Commentaire : Le parcours suit les voisines partageant la même couleur, en évitant les balises déjà visitées.*

Question 6

```
1 - 2 - 4 - 5 - 10 - 12 - 9 - 8 - 11 - 6 - 3 - 7
```

👉*Commentaire : Parcours en profondeur classique avec choix de la plus petite balise disponible à chaque étape.*

Question 7

```
balise4.voisines = [(balise2, 7), (balise5, 3), (balise6, 4)]
```

👉*Commentaire : On se réfère aux temps indiqués sur le graphe de la Figure 2.*

Question 8

Réponse : 5

👉*Commentaire : Parmi les voisines de balise10, c'est balise5 qui est la plus proche de la balise10.*

Question 9

```
1 - 2 - 4 - 6 - 11 - 9 - 12
```

👉*Commentaire : À chaque étape, le sportif choisit la balise non visitée accessible le plus rapidement.*

Question 10

```
def itineraire_trail(balise_debut, balise_fin):
    balise_debut.visitee = True
    balise = balise_debut
    chemin = [balise]
    while balise_fin not in chemin:
        prochaine = mystere(balise)
        if prochaine != None:
            prochaine.visitee = True
            chemin.append(prochaine)
            balise = prochaine
        else :
            return None
    return [b.num_balise for b in chemin]
```

👉*Commentaire : On utilise la fonction `mystere` pour choisir à chaque étape la meilleure prochaine balise non visitée.*

Question 11

Proposition A : algorithme glouton

👉*Commentaire : On fait un choix optimal local à chaque étape, sans considérer l'ensemble du chemin.*

Question 12

```
Avantage : rapide et simple à implémenter.
Inconvénient : ne garantit pas d'obtenir la solution optimale globale.
```

👉*Commentaire : L'approche gloutonne est efficace pour des solutions approximatives, mais ne fournit pas toujours la solution optimale.*
