Amérique du Nord – 2025 – sujet2 - Correction

Exercice 2 (6 points)

Partie A – Problème des bonbons

1. Cas de 8 bonbons

Les bonbons sont mangés successivement aux indices : 0, 3, 6, 2, 7, 5, 4.
Le bonbon restant est celui d'indice 1.

---

2. Erreur NameError

Une erreur NameError se produit lorsque Python rencontre un nom qui n'est pas défini.
Ici, true n'existe pas en Python. Il faut utiliser la valeur booléenne True avec une majuscule.
Instruction corrigée : collier = [True for i in range(8)].

---

3. Fonction dernier (liste de booléens)

```
def dernier(n):
    collier = [True for i in range(n)]
    reste = n
    i = 0
    collier[i] = False
    reste -= 1
    while reste > 1:
        pas = 0
        while pas < 3:
            i = (i + 1) % n
            if collier[i]:
                pas += 1
        collier[i] = False
        reste -= 1
    return collier.index(True)
```

👉*Commentaire : on parcourt circulairement la liste en sautant les bonbons déjà mangés jusqu'à atteindre le troisième.*

Partie B – Structure de file

4. Type de structure

La structure File correspond à l'acronyme FIFO : First In First Out.

---

5. Affichage obtenu

Après exécution des instructions, l'affichage est :
(Tête) 2 4 1 (Queue)

---

6. Fonction dernier_file

```
def dernier_file(n):
    f = File()
    for i in range(n):
        f.enfile(i)
    while f.taille() > 1:
        f.enfile(f.defile())
        f.enfile(f.defile())
        f.defile()
    return f.defile()
```

👉*Commentaire : on simule exactement le processus « manger un bonbon sur trois » avec une file.*

Partie C – Liste doublement chaînée

7. Vocabulaire POO

pred, valeur et succ sont des attributs de la classe `Bonbon`

---

8. Valeurs de a et b

a = zero.succ.valeur = 1
b = un.succ.succ.pred.valeur = 2

---

9. Fonction creer_collier

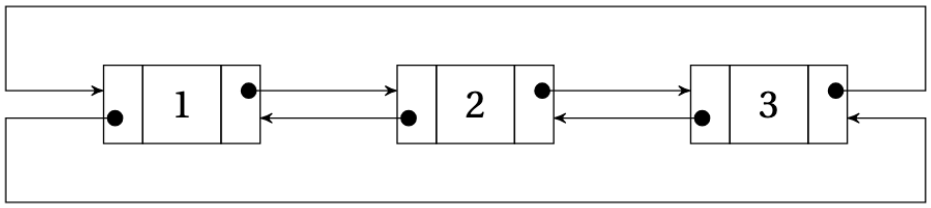
```
def creer_collier(n):
    premier = Bonbon(0)
    courant = premier
    for i in range(1, n):
        nouveau = Bonbon(i)
        courant.succ = nouveau
        nouveau.pred = courant
        courant = nouveau
    courant.succ = premier
    premier.pred = courant
    return premier
```

👉*Commentaire : on crée une liste doublement chaînée circulaire.*

---

10. Représentation du collier

bonbon est le 1. Le 0 a été enlevé du collier.



---

11. Condition lorsqu'il reste un seul bonbon

La bonne réponse est la proposition B : `pred == succ.`

---

12. Fonction dernier_chaine

```
def dernier_chaine(n):
    courant = creer_collier(n)
    while courant.succ != courant:
        courant.pred.succ = courant.succ
        courant.succ.pred = courant.pred
        courant = courant.succ.succ
    return courant.valeur
```

👉*Commentaire : la suppression d'un bonbon se fait en modifiant les liens prédécesseur et successeur.*