Combiner les outils de l’algorithmique

Module 9 • 60 minutes • BTS SIO1

Nom : _____ Date : _____

Le service informatique reçoit des demandes normales, notées "N", et urgentes, notées "U". Votre mission est de compter les urgences et de vérifier le résultat. Les lettres sont déjà en majuscules ; aucun nettoyage des données n’est demandé.

Cherchez d’abord, prévoyez les résultats, puis vérifiez en Python ou à la main. Le mémo séparé peut rester ouvert. Les aides sont en page 4 ; le défi de la page 5 est facultatif.

Reconnaître des outils déjà utilisés 6 min

Expliquez chaque extrait avec vos mots. Il n’est pas nécessaire de connaître le nom de l’outil. Les lignes sont des extraits indépendants ; les blocs des boucles et du if ne sont pas montrés.

Extrait Python	Que fait cet extrait ?
`nombre = 0`	
`if priorite == "U":`	
`for priorite in priorites:`	
`while stock < 3:`	
`nombre = compter_urgentes(priorites)`	

Comprendre un traitement complet 10 min

```
Fonction compter_urgentes(priorites : liste) : entier
    nombre ← 0
    Pour Chaque priorite dans priorites Faire
        Si priorite = "U" Alors
            nombre ← nombre + 1
        Fin Si
    Fin Pour
    Retourner nombre
Fin Fonction
```

Données : priorites ← ["N", "U", "N", "U"]. Complétez la trace.

Priorité lue	N	U	N	U
nombre après le passage				

Valeur retournée : _____. Entourez dans le code : initialisation, boucle, condition et retour. Pourquoi le retour est-il placé après la boucle ?

.....

.....

Compléter traduire et tester

Compléter le pseudo code 10 min

Reconstituez le même traitement, sans regarder la page précédente. Expliquez ensuite le rôle de chaque trou.

```
Fonction compter_urgentes(priorites : liste) : entier
    nombre ← ______
    Pour Chaque priorite dans ______ Faire
        Si ______ Alors
            nombre ← ______
        Fin Si
    Fin Pour
    Retourner ______
Fin Fonction
```

Traduire et appeler la fonction 12 min

Écrivez Python dans votre éditeur ou au dos. Ajoutez un appel qui affiche le nombre d'urgences de ["N", "U", "N", "U"]. Définir une fonction ne lance pas son traitement : il faut l'appeler.

.....

.....

.....

.....

.....

.....

Préparez les résultats attendus avant de vérifier. [] désigne une liste vide.

Liste testée	Résultat attendu	Résultat vérifié
["N", "U", "N", "U"]		
[]		
["U"]		
["N", "N"]		

Quel test vérifie le cas où la boucle ne fait aucun passage ? _____

Corriger puis adapter un algorithme

Déboguer une condition 7 min

Cette variante doit toujours compter les urgences, mais son résultat est faux.

```
def compter_urgentes(priorites):
    nombre = 0
    for priorite in priorites:
        if priorite != "U":
            nombre = nombre + 1
    return nombre
```

Pour ["N", "U", "N"], résultat voulu : _____ ; résultat produit : _____.
Corrigez une ligne. Expliquez pourquoi ["N", "U"] ne suffit pas à révéler l'erreur.

.....

.....

Créer en réutilisant une idée 12 min

Un message peut contenir des points d'exclamation. Écrivez compter_marques(texte), qui renvoie leur nombre. Exemple : "Alerte! Vérifier!" doit donner 2. La chaîne vide "" doit donner 0. Comptez les caractères "!", pas les mots.

Commencez par un pseudo-code, puis traduisez en Python au dos ou dans votre éditeur. Proposez un troisième test, avec son résultat attendu, avant de lancer le programme.

.....

.....

.....

.....

Mon test supplémentaire : _____ Résultat attendu : _____
Résultat obtenu : _____ Correction éventuelle : _____

Bilan individuel 3 min

Pour lire directement les valeurs d'une liste, j'écris : _____
Pour renvoyer une valeur depuis une fonction, j'utilise : _____
Je sais choisir mes outils : seul / avec le mémo / avec une aide.
Une chose que je sais expliquer maintenant : _____

Aides graduées du module 9

Prenez l'aide 1, essayez à nouveau, puis passez à la suivante seulement si nécessaire. Masquez le pseudo-code complet pendant votre premier essai.

Compter les urgences

Aide 1 : le compteur démarre à zéro et augmente uniquement quand une urgence est lue.

Aide 2 : parcourez les valeurs, comparez à "U", puis retournez le compteur après tous les passages. En Python, l'égalité se teste avec ==.

Aide 3 avec le pseudo code complet

```
Fonction compter_urgentes(priorites : liste) : entier
    nombre ← 0
    Pour Chaque priorite dans priorites Faire
        Si priorite = "U" Alors
            nombre ← nombre + 1
        Fin Si
    Fin Pour
    Retourner nombre
Fin Fonction
```

Compter les points d'exclamation

Aide 1 : une chaîne peut se parcourir comme une liste, caractère par caractère.

Aide 2 : reprenez le compteur d'urgences. Remplacez la liste par texte et le test par une comparaison avec "!".

Aide 3 avec le pseudo code complet

```
Fonction compter_marques(texte : chaîne) : entier
    nombre ← 0
    Pour Chaque caractere dans texte Faire
        Si caractere = "!" Alors
            nombre ← nombre + 1
        Fin Si
    Fin Pour
    Retourner nombre
Fin Fonction
```

Pour vérifier : suivez les caractères un à un. Les espaces et les autres signes ne font pas augmenter le compteur. Pour un texte vide, le compteur reste à zéro.

Défi du module 9 – Prépa CCF

Environ 10 minutes • Réserver des créneaux aux urgences

Le service dispose de 2 créneaux. Il parcourt les priorités dans l'ordre. Une demande "U" prend un créneau s'il en reste ; sinon, elle reste en attente. Une demande "N" ne prend aucun créneau dans ce traitement. Affichez à la fin les créneaux restants et le nombre d'urgences en attente.

Compléter puis traduire

```
priorites ← ["N", "U", "U", "U"]
places ← 2
attente ← 0
Pour Chaque priorite dans priorites Faire
    Si __________ Alors
        Si __________ Alors
            places ← __________
        Sinon
            attente ← __________
        Fin Si
    Fin Si
Fin Pour
Afficher places, attente
```

Prévoyez les deux résultats pour cette liste, puis pour [] : _____
Traduisez dans votre éditeur. Repartez de places = 2 à chaque test.

Aides graduées

Aide 1 : seules les urgences changent quelque chose.

Aide 2 : parmi les urgences, distinguez places > 0 et absence de place.

Aide 3 avec le pseudo code complet

```
places ← 2 ; attente ← 0
Pour Chaque priorite dans priorites Faire
    Si priorite = "U" Alors
        Si places > 0 Alors
            places ← places - 1
        Sinon
            attente ← attente + 1
        Fin Si
    Fin Si
Fin Pour
Afficher places, attente
```