Répéter des instructions

Module 4 • 60 minutes • BTS SIO1 • Nom : _____ Date : _____

Votre mission : automatiser des *opérations répétées au service informatique. Vous utiliserez for quand le nombre de passages est connu, puis découvrirez while quand une condition commande la répétition. Sans poste, suivez les valeurs sur papier. Mémo et aides : page 4.*

Réactiver une mise à jour 5 min

Après stock ← 2 puis stock ← stock + 1, stock vaut _____.

Si l'on exécute encore stock ← stock + 1, stock vaut _____.

Comprendre une boucle bornée 10 min

Une livraison contient 4 claviers. Chaque passage ajoute un clavier au stock. Pour numero de 1 à 4 signifie quatre passages, avec la variable numero valant successivement et automatiquement de 1, 2, 3, 4.

```
stock ← 2
Pour numero de 1 à 4 Faire
    stock ← stock + 1
Fin Pour
Afficher stock
```

Complétez la trace. La ligne Afficher stock se trouve après la boucle.

Moment	numero	stock
Avant la boucle	Non défini	2
Après le passage 1	1	
Après le passage 2	2	
Après le passage 3	3	
Après le passage 4	4	

Combien de fois l'ajout est-il exécuté ? _____

Combien de fois le stock est-il affiché ? _____ Valeur affichée : _____

Compléter une répétition 6 min

Il reste 6 souris. On en distribue 3, une par passage. Complétez puis prévoyez la valeur finale.

```
stock ← 6
Pour numero de 1 à _____ Faire
    stock ← __________
Fin Pour
Afficher stock
```

Résultat prévu : _____. Expliquez le rôle de la ligne dans la boucle.

.....

.....

Passer à for en Python

Comprendre i et traduire 10 min

numero et i sont deux noms possibles pour la variable de boucle. i est un nom courant, pas un mot imposé par Python. On garde notre stock de 2 claviers et on effectue quatre ajouts :

```
stock = 2
for i in range(4):
    stock = stock + 1
print(stock)
```

range(4) donne 0, 1, 2, 3 : le départ est 0 par défaut et la borne finale 4 est exclue. Complétez avant de tester.

Passage	Valeur de i	stock après l'ajout
Premier	0	
Deuxième	1	
Troisième	2	
Quatrième	3	

Nombre de passages : _____ ; stock final : _____. Le passage où i vaut 0 compte bien : le bloc est exécuté une première fois. i n'est pas le stock.

Repartez de stock = 2 et remplacez seulement range(4) par range(1, 5). Valeurs de i : _____. Le stock final change-t-il ? _____. Expliquez oralement.

Traduisez maintenant la distribution de 3 souris (page 1), avec for i in range(3). Affichez le stock après la boucle. Écrivez au dos ou dans votre éditeur. Stock final prévu : _____.

Déboguer deux erreurs 7 min

On veut afficher 1, 2, 3, 4, puis le mot « Terminé » une seule fois. Le code suivant ne le fait pas.

```
for i in range(1, 4):
    print(i)
    print("Terminé")
```

1. Écrivez les sorties actuelles dans leur ordre.
2. Corrigez la borne et la place du dernier print.

.....

.....

.....

.....

Créer une répétition 9 min

Une réserve contient 10 câbles. Quatre postes reçoivent chacun un câble. Écrivez le pseudo-code puis Python pour retirer un câble à chaque passage. Affichez uniquement le stock final. Sur le dos ou dans votre éditeur, montrez aussi les quatre valeurs successives du stock pour vérifier votre programme.

Stock final prévu : _____. Stock final vérifié : _____.
Le nombre de passages se décide ici avant la boucle : oui / non.

Vous allez maintenant comparer cette boucle avec une répétition commandée par une condition.

Découvrir while avec une condition

Comprendre et compléter 10 min

On ajoute un clavier à chaque passage tant que le stock est inférieur à 5. Avec while, la condition est testée avant chaque passage. Le programme peut donc effectuer zéro passage.

```
stock ← 2
Tant Que stock < 5 Faire
    stock ← stock + 1
Fin Tant Que
Afficher stock
```

```
stock = 2
while stock < 5:
    stock = stock + 1
print(stock)
```

1. Complétez la trace sans exécuter. Le dernier test permet de sortir de la boucle.

stock avant le test	stock < 5 est vraie ou fausse	Action ou arrêt
2		
3		
4		
5		

2. Nombre de passages : _____. Valeur affichée : _____.

3. Si stock vaut 5 au départ, combien de passages effectue-t-on ? _____

4. Complétez cette variante pour atteindre 6 en partant de 4.

```
stock = 4
while __________:
    stock = __________
print(stock)
```

5. Que risque-t-il de se passer si l'on remplace stock = stock + 1 par print(stock) dans la première boucle ? Pourquoi ?

.....

.....

Si une boucle ne s'arrête pas dans votre éditeur, utilisez son bouton Arrêter ou Interrompre. Avant de la relancer, vérifiez la condition et la mise à jour.

Bilan individuel 3 min

range(3) donne les valeurs _____ ; cela fait _____ passages.
Dans while, la condition est vérifiée avant / après chaque passage.
Pour qu'une boucle while puisse s'arrêter, je dois surveiller _____.
Je sais suivre une boucle à la main : oui / avec aide / pas encore.

Mémo et aides de la fiche 4

Choisir une répétition

Situation	Pseudo-code	Python
3 passages prévus	Pour i de 0 à 2 Faire	for i in range(3):
Répéter tant que le stock est trop bas	Tant Que stock < 5 Faire	while stock < 5:
Fin du bloc répété	Fin Pour ou Fin Tant Que	Revenir à l'alignement de départ

Une boucle répète toutes les instructions de son bloc. Une instruction placée après le bloc ne fait pas partie de la répétition. Avec for, range fixe ici les passages. Avec while, une condition décide de chaque nouveau passage.

Rappels utiles

range(3) donne 0, 1, 2 et range(1, 4) donne 1, 2, 3 : dans les deux cas, trois passages. i peut remplacer numero : seul le nom change. i prend les valeurs du parcours ; stock change selon les instructions. Ajouter ou retirer 1 modifie le stock d'une unité.

Aides graduées

Activité	Aide 1	Aide 2
Comprendre for	Faites une ligne par passage.	Partez de 2 ; ajoutez 1 quatre fois.
Compléter	Trois souris sortent une par une.	Borne : 3 ; mise à jour : stock - 1.
Traduire	Avec un paramètre, le départ est 0.	Trois passages : range(3). Le stock reste mis à jour dans le bloc.
Déboguer	Comptez les valeurs de range.	La borne doit être 5 ; Terminé sort du bloc.
Créer	Quatre postes = quatre passages.	Partez de 10 et retirez 1 à chaque passage.
Comprendre while	Testez, mettez à jour, retestez.	À 5, stock < 5 est faux ; on s'arrête.

Aide 3 pour traduire avec i

```
stock ← 6
Pour i de 0 à 2 Faire
    stock ← stock - 1
Fin Pour
Afficher stock
```

Aide 3 pour compléter while

```
stock ← 4
Tant Que stock < 6 Faire
    stock ← stock + 1
Fin Tant Que
Afficher stock
```

Défi facultatif : page 5. Extra sur range à un, deux ou trois paramètres : page 6. Ces prolongements ne s'ajoutent pas au parcours obligatoire de 60 minutes.

Défi final de la fiche 4

Facultatif • Environ 10 minutes • Traiter cinq demandes

Le service possède 3 casques. Cinq personnes demandent chacune un casque. Pour chaque personne, si le stock est supérieur à 0, retirer un casque et afficher « Prêt accepté ». Sinon, afficher « Prêt refusé » sans changer le stock. Afficher le stock final une seule fois après les cinq demandes.

Compléter puis traduire

```
stock ← 3
Pour numero de 1 à ______ Faire
    Si __________ Alors
        stock ← __________
        Afficher "Prêt accepté"
    Sinon
        Afficher "Prêt refusé"
    Fin Si
Fin Pour
Afficher __________
```

1. Complétez puis traduisez en Python sur le dos ou dans votre éditeur.
2. Prévoyez le nombre de prêts acceptés, de refus et le stock final.
3. Testez aussi avec stock = 5, puis stock = 0 au départ. Gardez cinq demandes.

Stock initial	Prêts acceptés	Refus	Stock final
3			
5			
0			

Aides graduées pour le défi

Aide 1 : une personne correspond à un passage ; la décision se prend dans la boucle.

Aide 2 : utilisez for i in range(5) et testez stock > 0. Le if est décalé de quatre espaces et ses instructions de huit espaces. Le pseudo-code numérote ici de 1 à 5 ; range(5) parcourt 0 à 4. Cinq demandes sont traitées dans les deux cas car le numéro n'est pas utilisé dans le calcul.

Aide 3 Pseudo-code complet

```
stock ← 3
Pour numero de 1 à 5 Faire
    Si stock > 0 Alors
        stock ← stock - 1
        Afficher "Prêt accepté"
    Sinon
        Afficher "Prêt refusé"
    Fin Si
Fin Pour
Afficher stock
```

Extra explorer les paramètres de range

Facultatif • Environ 8 minutes • Pour les plus rapides ou lors d'une reprise

Un deux ou trois paramètres

Les paramètres de range sont séparés par des virgules. Avec un paramètre, on donne la borne finale. Avec deux, le départ puis la borne finale. Avec trois, on ajoute le pas : la quantité ajoutée à i entre deux valeurs. La borne finale reste exclue.

Écriture	Valeurs successives de i	Passages
range(4)	0, 1, 2, 3	4
range(1, 5)	1, 2, 3, 4	4
range(2, 9, 2)	2, 4, 6, 8	4

Prévoir puis vérifier

```
for i in range(2, 9, 2):
    print(i)
```

1. Identifiez le départ : _____ ; la borne exclue : _____ ; le pas : _____.
2. Pourquoi 9 n'est-il pas affiché ? _____
3. Que devient l'affichage si on utilise range(2, 8, 2) ? _____
4. Sans exécuter, complétez puis vérifiez :

Écriture	Valeurs prévues	Passages
range(5)		
range(3, 7)		
range(1, 8, 2)		
range(4, 4)		

Petit défi avec un pas

Affichez les numéros 2, 4, 6, 8 et 10 avec une seule boucle. Complétez d'abord ce pseudo-code, puis traduisez-le en Python.

```
Pour i de _____ à _____ avec un pas de _____ Faire
    Afficher i
Fin Pour
```

.....

.....

Aides graduées

Aide 1 : commencez à 2 et avancez de 2.

Aide 2 : la dernière valeur souhaitée est 10 ; choisissez une borne Python exclue qui permet de l'atteindre.

Aide 3 : Pour i de 2 à 10 avec un pas de 2 Faire ; Afficher i ; Fin Pour. La borne du pseudo-code est ici incluse.

À retenir : le pas de range ne modifie pas automatiquement stock. Seules les instructions du bloc le font. Les pas négatifs seront étudiés plus tard.