UAA4 et UAA5 – Python

Apprentissage à la programmation impérative par des Travaux Pratiques et des Projets

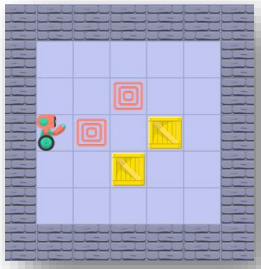
1 TP1 – Suites d'instructions

1.1 Algorea-1 : Découvrir comment donner des ordres à un robot

⇨ https://parcours.algorea.org/contents/4707-4702-1471479157476024035

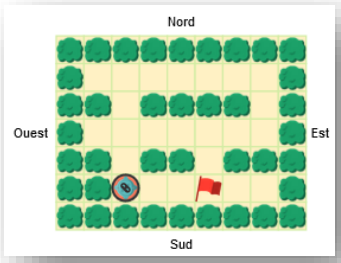
1. Algo - Pousser les caisses

VERSION ★★★★ (C)



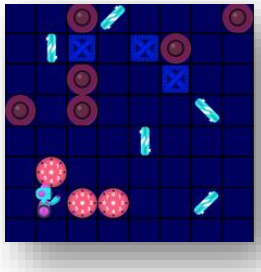
2. Algo - Trouver la sortie

VERSION ★★★★ (C)

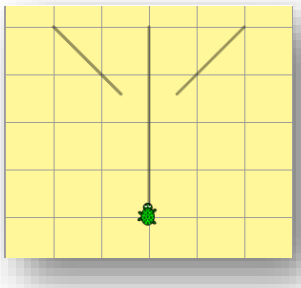


3. Algo - Tirer au laser

VERSION ★ VERSION ★★★★ (M)



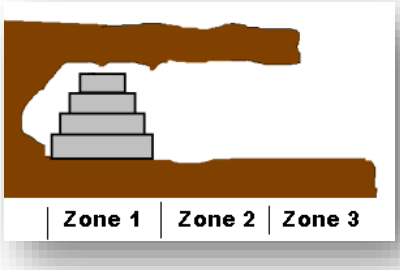
Algo - Dessiner avec la tortue VERSION ★★★★ (M) Challenge



1.2 Fce-ioi-1 : Affichage de texte et Suite d'instructions

⇨ https://www.france-ioi.org/algo/chapters.php

1. Fce-io 1.1 – 1) Hello word (C) Découverte
2. Fce-io 1.1 – 2) Présentation (C) Entraînement
3. Fce-io 1.1 – 5) Empilement de cylindres (M) Challenge



4. Fce-io 1.1 – 6) Recette secrète (M) Challenge

5. Sauvegarde (C)

Sur ton PC, sous **uaa5_python_prenom**, crée un dossier **tp1**, puis **tp1-1**
A la fin de chaque exercice:

- Copie ton code (de France-IOI ou Algorea) dans un fichier « ***testN_titre-de-l'exerice.py*** »
- Sauvegarde tes fichiers sur le drive dans la rubrique zone_transfert_eleves_Info4TTi / ton_prenom:

2 TP2 – Répétitions d'instructions: les boucles

2.1 Algorea-2 : Donner plusieurs fois le même ordre au robot

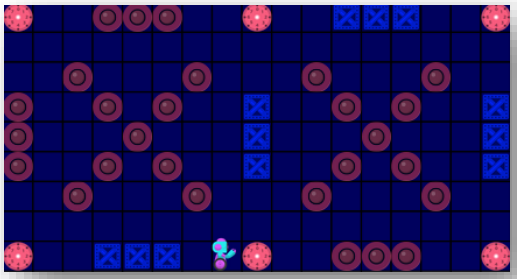
⇨ https://parcours.algorea.org/contents/4707-4702-1471479157476024035

1. Algo - Planter des fleurs

VERSION ★★★★ (C)

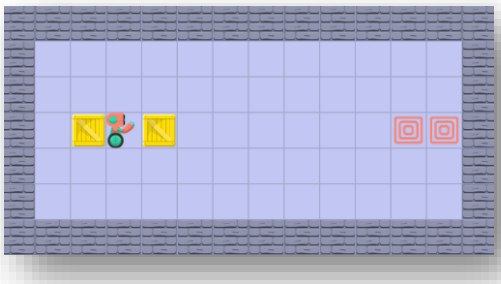
2. Algo - Tir au laser

VERSION ★★★★ (M)



3. Algo - Pousser les caisses

VERSION ★★★★ (C)



2.2 Fce-ioi-2 : Répétitions d'instructions

⇨ https://www.france-ioi.org/algo/chapters.php

Synthèse de cours

⇨ **Lis en complément le petit cours suivant :**
https://e-nsi.forge.aeif.fr/init_python/2-repeat/14-repeat

1. **Fce-ioi 1.2 1) Punition** (C) Découverte

(afficher un grand nombre de fois le même texte)

2. **Fce-ioi Répétition: erreurs possible** (C) Cours

3. **Fce-ioi Indentation: la touche tabulation** (C) Cours

4. **Fce-ioi 1.2 2) Mathématiques de base** (M) Entraînement

(corriger les erreurs de syntaxes d'un programme)

⇨ À exécuter sous idle

5. **Fce-ioi 1.2 3) Transport d'eau** (C) Entraînement

6. **Fce-ioi 1.2 4) Le secret du Goma** (M) Découverte

7. **Fce-ioi Répétition: cohérence de l'indentation** (C) Cours

8. **Fce-ioi 1.2 5) Sisyphe:** (M) Entraînement

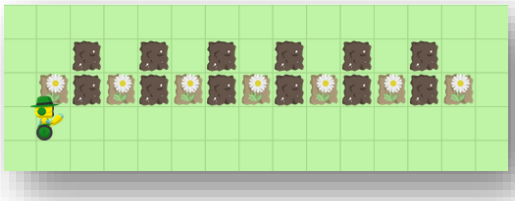
9. **Fce-ioi 1.2 – 6) Page d'écriture** (C) Découverte

2.3 Algorea-3 : Donner plusieurs fois la même séquence d'ordre au robot

⇨ https://parcours.algorea.org/contents/4707-4702-1471479157476024035

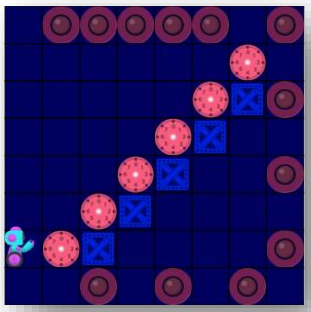
1. Algo - Planter des fleurs

VERSION ★★★★ (C)



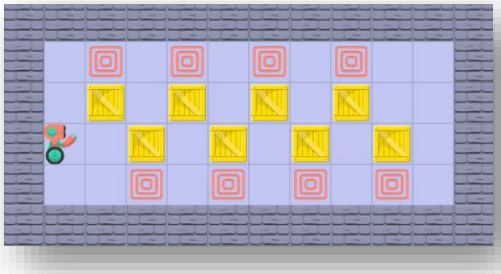
2. Algo - Tirer au laser

VERSION ★★★★ (M)



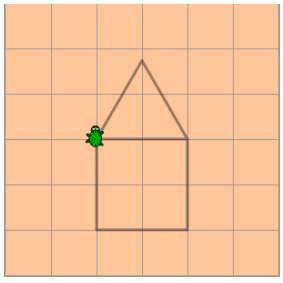
3. Algo - Pousser les caisses

VERSION ★★★★ (C)



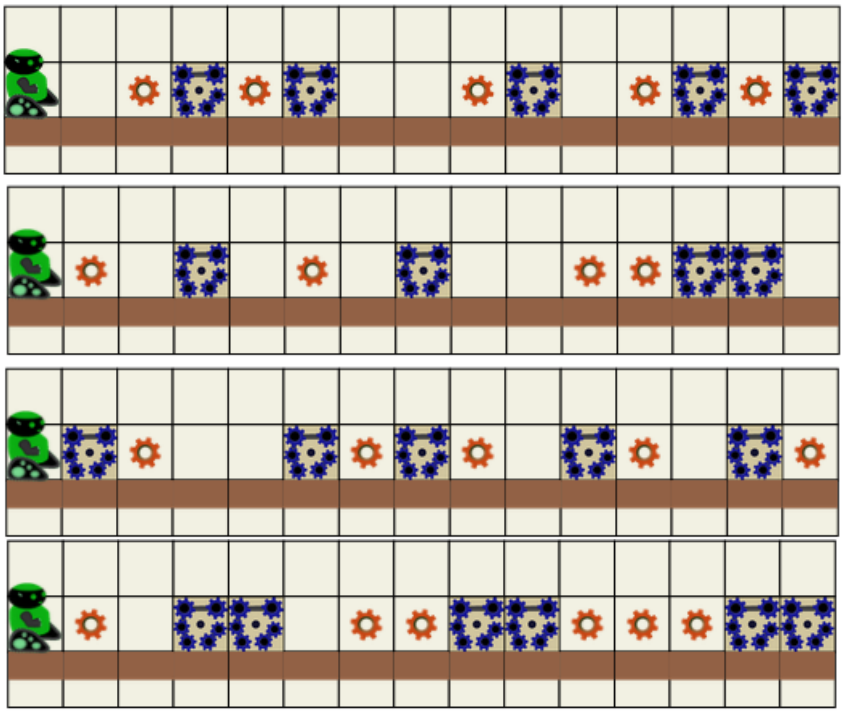
4. Algo - Dessiner avec la tortue

VERSION ★★★★ (M)



2.4 Algo-8 : Créer ses propres blocs : Introduction aux fonctions

1. Algo – Construire une machine



VERSION ★ (C)

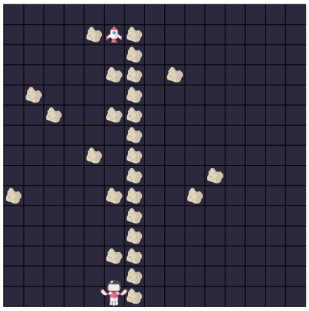
VERSION ★★ (C)

VERSION ★★★ (C)

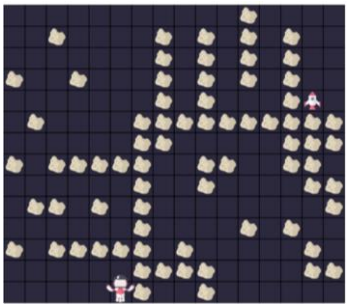
VERSION ★★★★ (M) Challenge

2. Algo – Rejoindre la fusée

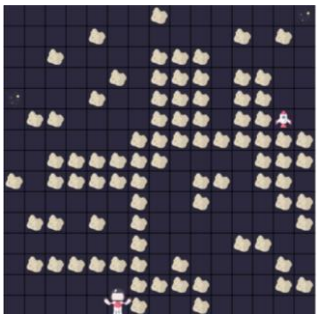
VERSION ★★ (C)



VERSION ★★★ (C)



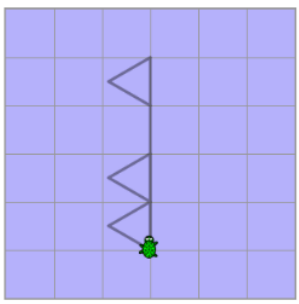
VERSION ★★★★ (M) Challenge



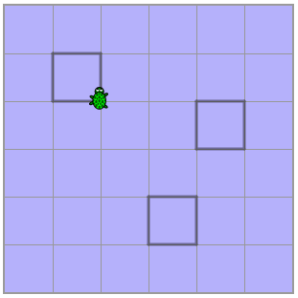
3. Algo – Dessiner avec la tortue

⇨ Indice : dessine la figure comme 8 rotations du petit carré

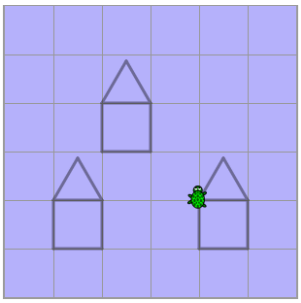
VERSION ★ (C)



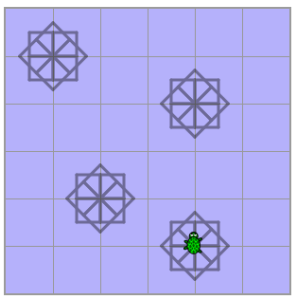
VERSION ★★ (C)



VERSION ★★★ (C)



VERSION ★★★★ (C)



2.5 Algorea-4 : Concevoir des programmes compacts : Boucles imbriquées

⇨ https://parcours.algorea.org/contents/4707-4702-1471479157476024035

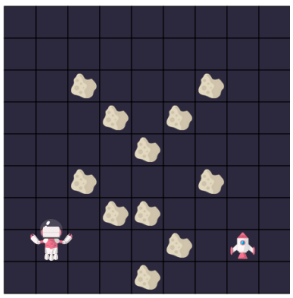
⇨ Bien se représenter sur le dessin la partie qui se répète

1. Algo - Rejoindre la fusée

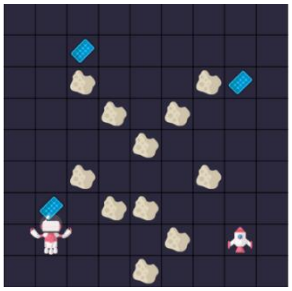
⇨ Repère sur la figure le déplacement qui se répète

⇨ La VERSION ★★★★ peut être réalisée en 10 instructions (ou blocs)

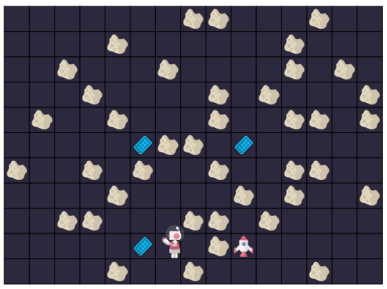
VERSION ★★ (C)



VERSION ★★★ (C)

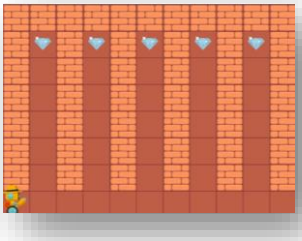


VERSION ★★★★ (C)



2. Algo - Collecter les pierres précieuses

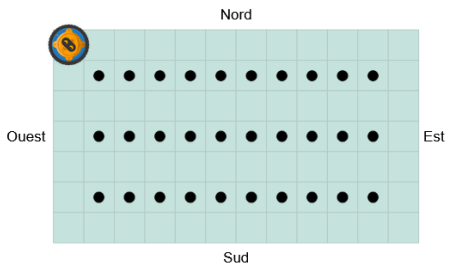
VERSION ★★★★ (C)



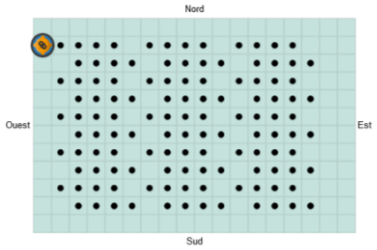
3. Algo - Peindre le motif

⇨ La VERSION ★★★★ peut être réalisée en 13 instructions (ou blocs)

VERSION ★★★



VERSION ★★★★ (C)



2.6 Algorea-Turtle : boucles bornées et boucles imbriquées

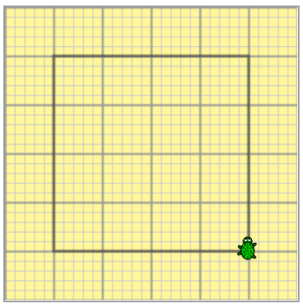
⇨ https://old.parcours.algorea.org/contents/4707-4702-1352246428241737349-314613032161178344-310572474623192846

Programmez votre tortue pour qu'elle se déplace sur le trait gris et y peigne sa trace.
La grille est un carré de 300 pixels de côté. Un pas de tortue correspond à un pixel.
Fonctions disponibles:

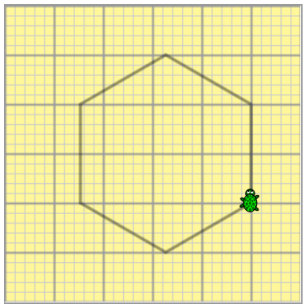
⇨ foward(steps), backward(steps), left(angle), right(angle), color(colorName)

1. Algo-Turtle : Les boucles bornées ou boucle à compteur

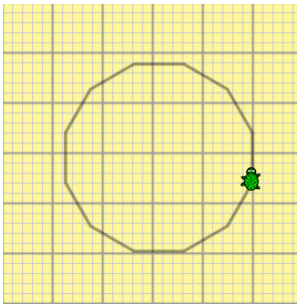
VERSION ★(M)



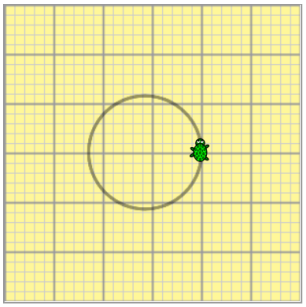
VERSION ★★(M)



VERSION ★★★(M)

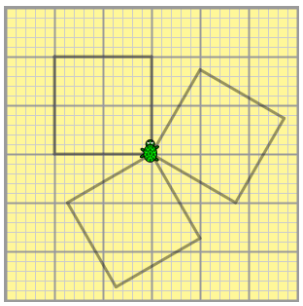


VERSION ★★★★ (M) Challenge

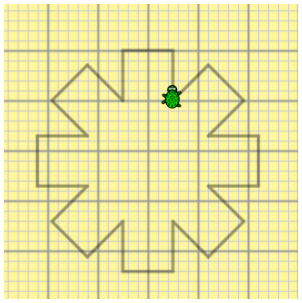


2. Algo-Turtle : Les boucles imbriquées:

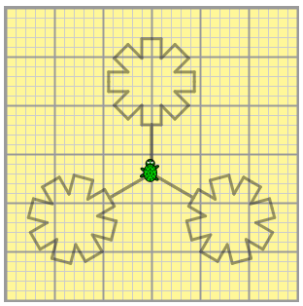
VERSION ★(M)



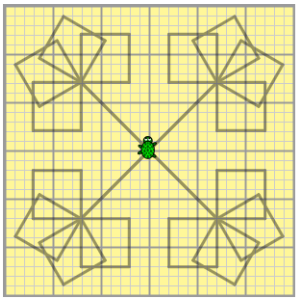
VERSION ★★(M)



VERSION ★★★ (M) Challenge



VERSION ★★★★★ (M) Challenge



2.7 Fce-ioi-2 : Répétitions d'instructions (suite – boucles imbriquées)

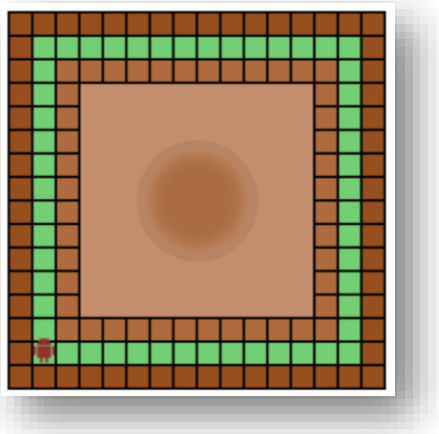
⇨ https://www.france-ioi.org/algo/chapters.php

1. Fce-ioi 1.2 – 7) Jeu de dame (C) Découverte

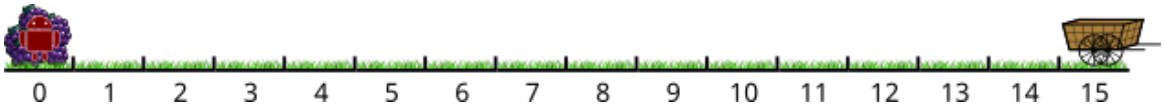
(imbriquer des répétitions pour dessiner un tableau)

⇨ **À exécuter sous idle**

2. Fce-ioi 1.2 - 8) Mont Kailash (M) Entraînement



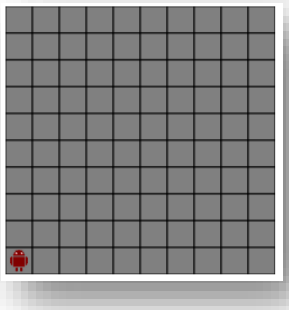
3. Fce-io 1.2 – 9) Vendages (C) Validation



(faire faire une tâche au robot avec des répétitions imbriquées)

4. Fce-ioi Insérer des commentaires (C) Cours

5. Fce-io – 10) Le grand évènement (M) Challenge



6. Fce-ioi Bien lire les corrections (C) Cours

2.8 Idle : Boucles imbriquées - Synthèse

1. Idle – Synthèse 1 (C)

Ce que doit faire ton programme

Ecris 5 fois « Bonjour » suivi de 5 fois par « Comment vas-tu ? », chaque mot ne pouvant apparaître qu'une seule fois dans le programme.

Sortie

```
Bonjour !
Bonjour !
Bonjour !
Bonjour !
Bonjour !
Comment vas-tu ?
Comment vas-tu ?
Comment vas-tu ?
Comment vas-tu ?
Comment vas-tu ?
```

2. Idle – Synthèse 2 (C)

Ce que doit faire ton programme

Ecris 5 fois: « Bonjour » « Comment vas-tu ? »

Sortie

```
Bonjour !
Comment vas-tu ?
Bonjour !
Comment vas-tu ?
Bonjour !
Comment vas-tu ?
Bonjour !
Comment vas-tu ?
Bonjour !
Comment vas-tu ?
```

3. Idle – Synthèse 3 (C)

Ce que doit faire ton programme

Ecris 5 fois: 4 « Bonjour » suivi de 2 « Comment vas-tu ? »

Sortie

```
Bonjour !
Bonjour !
Bonjour !
Bonjour !
Comment vas-tu ?
Comment vas-tu ?
Bonjour !
Bonjour !
Bonjour !
Bonjour !
Comment vas-tu ?
Comment vas-tu ?
Bonjour !
Bonjour !
```

```
Bonjour !
Bonjour !
Comment vas-tu ?
Comment vas-tu ?
Bonjour !
Bonjour !
Bonjour !
Bonjour !
Comment vas-tu ?
Comment vas-tu ?
Bonjour !
Bonjour !
Bonjour !
Bonjour !
Comment vas-tu ?
Comment vas-tu ?
```

4. Idle – Synthèse 4 (C)

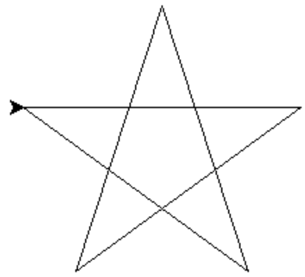
Ce que doit faire ton programme

Ecris le code donnant le résultat ci-dessous.

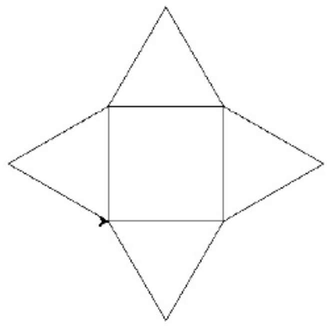
Sortie

```
Bonjour ! Comment vas-tu ? Comment vas-tu ? Comment vas-tu ? Comment vas-tu ? Bonjour !
Tu réponds ????? ????? ????? ????? ????? ????? ????? ????? ????? ?????
Tu réponds ????? ????? ????? ????? ????? ????? ????? ????? ????? ?????
Tu réponds ????? ????? ????? ????? ????? ????? ????? ????? ????? ?????
Tu réponds ????? ????? ????? ????? ????? ????? ????? ????? ????? ?????
Tu réponds ????? ????? ????? ????? ????? ????? ????? ????? ????? ?????
Tu réponds ????? ????? ????? ????? ????? ????? ????? ????? ????? ?????
Tu réponds ????? ????? ????? ????? ????? ????? ????? ????? ????? ?????
Tu réponds ????? ????? ????? ????? ????? ????? ????? ????? ????? ?????
Tu réponds ????? ????? ????? ????? ????? ????? ????? ????? ????? ?????
Bonjour ! Comment vas-tu ? Comment vas-tu ? Comment vas-tu ? Comment vas-tu ? Bonjour !
```

5. Idle – Turle, Synthèse 5 (M)



6. Idle – Turle, Synthèse 6 (M)



1. Idle – Synthèse 7 (C)

Ce que doit faire ton programme

Ecris le code donnant le résultat ci-dessous.

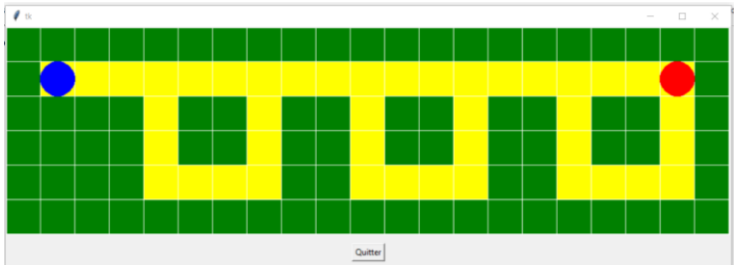
Sortie

```
Bonjour !
Comment vas-tu ? Comment vas-tu ? Comment vas-tu ? Comment vas-tu ?
Tu réponds ?
Comment vas-tu ? Comment vas-tu ? Comment vas-tu ? Comment vas-tu ?
Je te parle !
Comment vas-tu ? Comment vas-tu ? Comment vas-tu ? Comment vas-tu ?
Tu ne m'entends pas !
Comment vas-tu ? Comment vas-tu ? Comment vas-tu ? Comment vas-tu ?
Bon, ben, salut ...
```

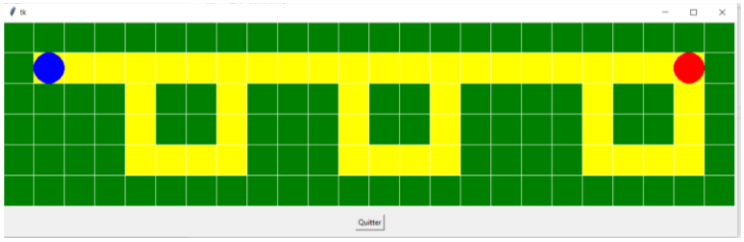
2. Idle – Robot synthèse tp2

- ✓ Télécharge du drive le dossier **tp2_algorea_synthese**, décompresse les fichiers et déplace-les dans ton dossier de travail.
- ✓ Modifie et exécute le fichier **test.py**. L'appel des test1, 2, 3 se fait en modifiant l'instruction en fin de code test1() par test2() ou test3()
- ✓ Déplace le point bleu jusqu'au point rouge en passant par toutes les cases jaunes.
  Tu dois utiliser un minimum d'instructions, le déplacement se fait grâce à :
  est(), ouest(), nord(), sud()
- ✓ Démarche :
  - Pense à bien repérer sur la figure le bout de dessin qui se répète.
  - Ecris tout d'abord le code pour parcourir ce bout de dessin.
  - Complète-le pour parcourir toute la figure.

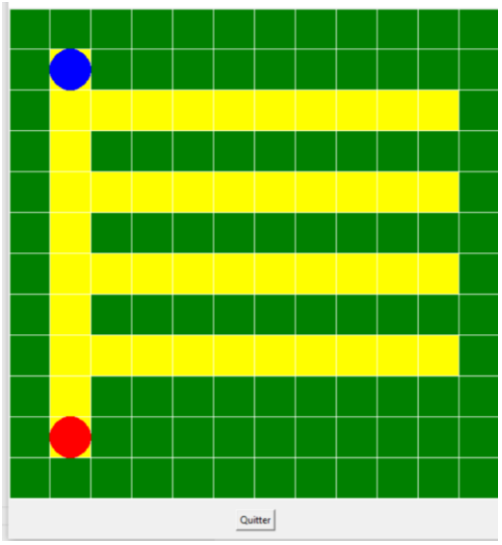
test1



test2



test3



3. Mémo de synthèse

- ✓ A partir des exercices que tu as réalisés, écris un mémo des instructions python que tu as rencontrées.

3 TP3 – Les variables

Consignes (rappel)

A la fin de chaque exercice:

- Sur Fce-ioi ou algorea: copie ton code dans un fichier ***tp3_titre-de-l'-exerice.py***
- Livre tes fichiers sur le drive dans la rubrique:
  zone_transfert_eleves_Info4TTi -> xx_ton-prenom -> date -> tp3

3.1 Fce-ioi-3 / Idle : Calculs et découverte des variables : découverte

1. Idle Afficher un entier (C) Découverte

✓ Ouvre un terminal, crée un dossier tp3, crée un fichier xx_tp3_entier.c

✓ Ecris le code suivant, compile-le et exécute-le:

```
print("2+3")
```

Sortie

```
2+3
```

L'ordinateur affiche à l'écran le **texte** 2+3

✓ Ecris le code suivant:

```
print(2+3)
```

Sortie

```
5
```

L'ordinateur a affiché à l'écran le **nombre** 5 qui est le résultat du calcul 2+3

✓ Ecris le code suivant:

```
print( ' la somme de 2 avec 3 est ' , 2+3 , ' il est fort cet ordinateur !')
```

Sortie

```
la somme de 2 avec 3 est 5, il est fort cet ordinateur !
```

Le résultat du calcul 2+5 est inséré dans la zone de texte à l'emplacement du %

2. **Fce-ioi 1.3 2) L'éclipse** (C) Découverte

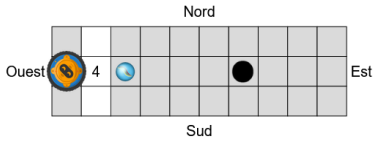
   (opérateurs arithmétiques)

3. **Fce-ioi 1.3 3) Bondons pour tout le monde** (M) Entraînement

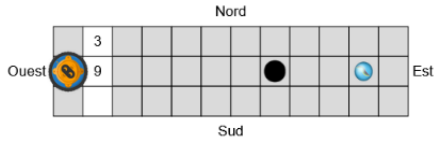
   (opérateurs arithmétiques)

3.2 Algorea-6 : Garder de l'information en mémoire

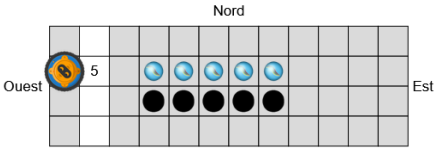
1. Ranger les billes



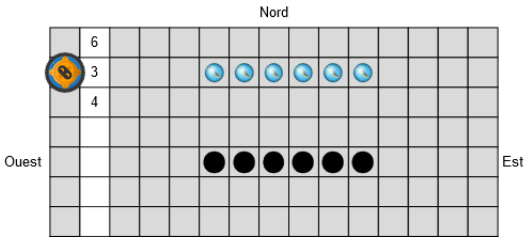
VERSION ★ (C)



VERSION ★★ (C)



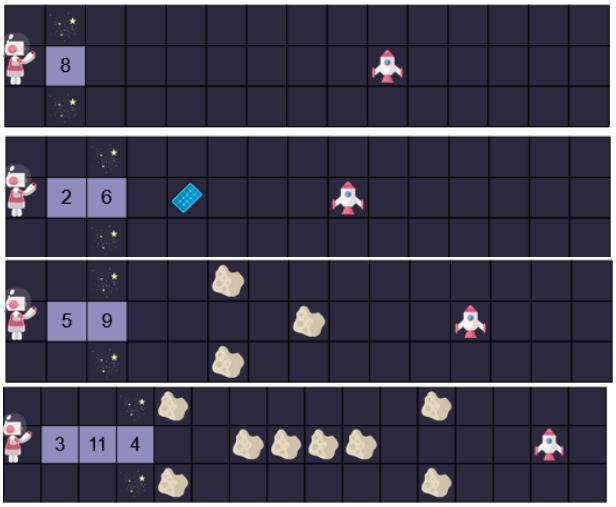
VERSION ★★★ (C)



VERSION ★★★★ (M)

Challenge

2. Rejoindre la fusée



VERSION ★ (C)

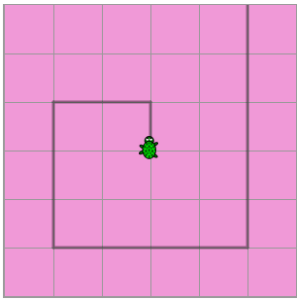
VERSION ★★ (C)

VERSION ★★★ (C)

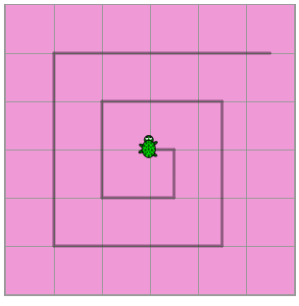
VERSION ★★★★ (M)

3. Dessiner avec la tortue

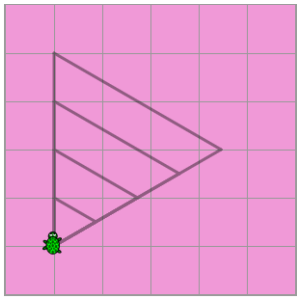
VERSION ★ (C)



VERSION ★★ (C)

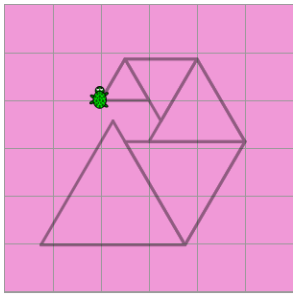


VERSION ★★★ (C)



VERSION ★★★★ (M)

Challenge



3.3 Fce-ioi-3 / Idle : Calculs et découverte des variables : compteurs

1. Idle / Fce-ioi 1.3 4) L'algoréathlon (C) Découverte

Qu'est-ce qu'une variable ?

- ✓ Dans un programme informatique, on a besoin en permanence de stocker provisoirement des valeurs. Il peut s'agir de données issues du disque dur, fournies par l'utilisateur (frappé au clavier), ou encore de résultats obtenus par le programme. Dès qu'on a besoin de stocker une information, on utilise une variable.
- ✓ On peut donc définir une variable comme une case mémoire qui permet de stocker une information.
- ✓ Elle est constituée de 2 éléments:
  - Une valeur: ici, c'est le nombre qu'on stocke.
  - Un nom qui permet de la reconnaître, l'ordinateur se charge d'associer à cette variable l'adresse d'une case mémoire dans la RAM.
- ✓ L'ordinateur a accès au contenu de cette variable pendant tout le déroulement du programme.
- ✓ Toutes les variables doivent être déclarées dès le début du programme en précisant le type d'information stockée (un entier, un réel, un caractère …), son nom, sa valeur initiale.
  On donne une valeur à la variable par l'**instruction d'affection « = »**
  Exemple, déclaration d'une variable qui représente le nombre d'euros disponible un compte en banque et affectation à 0:

```
nb_euro=0 ;
```

- ✓ Le « = » ne modifie que la variable qui est à gauche de la flèche.
  Exemple:

```
toto=12 ;
titi = 16 ;
titi = toto + 8
```

titi vaut maintenant 20 mais toto vaut toujours 12.

Remarque importante

- ✓ Attention, une valeur en informatique n'est pas la valeur que vous connaissez des mathématiques !!!
- ✓ En mathématiques, une variable est généralement une inconnue, qui recouvre un nombre non précisé de valeurs.
  Lorsqu'on écrit:
  $y = 3x + 2$,
  les variables x et y satisfaisant l'équation existent en nombre infini (le graphe des solutions correspond à une droite).
  Lorsqu'on écrit:
  $ax^2 + bx + x = 0$
  la variable x désigne les solutions à cette équation et il peut y en avoir 0, 1 ou 2 valeurs.
  - ⇨ **En informatique, une variable possède à un instant donné une et une seule valeur** et cette valeur ne varie pas, du moins tant qu'elle ne fait pas l'objet d'une instruction d'affectation (instruction « = »)
- ✓ En mathématiques, A = B et B = A sont strictement équivalent.
  - ⇨ En informatique A = B et B = A sont deux choses différentes
- ✓ En mathématiques A = A + 1 est une équation sans solution
  - ⇨ **En informatique, A = A + 1 signifie que le contenu de A est incrémenté (augmenté) de 1.**

Synthèse de cours

- ⇨ **Lis en complément le petit cours suivant :**
  https://e-nsi.forge.aeif.fr/init_python/3-var/18-variable

Exercice

- ✓ Crée une variable contenant ton âge et affiche-là.
- ✓ Affiche-la 5 fois en lui rajoutant 1 à chaque fois.
- ✓ Affiche-la 5 fois en la multipliant par 2 à chaque fois.

Sortie

```
15
16
17
18
19
20

30
60
120
240
480
```

2. **Fce-ioi 1.3 5) Cours de récréation** (C) Entraînement

   (mesure de sa longueur avec des bâtons puis calcul du périmètre et de l'aire)

3. **Fce-ioi 1.3 6) Une partie de cache-cache** (C) Découverte

   (compter jusqu'à 100 dans une boucle)

4. **Fce-ioi Variables: suppléments** (C) Cours

5. **Fce-ioi 1.3 7) Progresser par l'erreur** (M) Entraînement

   (corriger les erreurs de syntaxes d'un programme)

6. **Fce-ioi 1.3 8) Décollage de fusée** (C) Entraînement

   (décompter à partir de 100 dans une boucle)

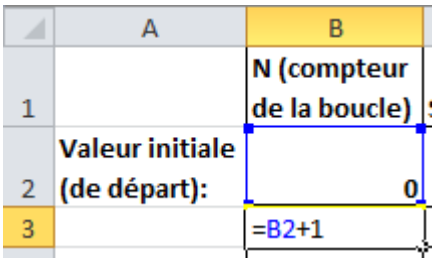
7. **Fce-ioi 1.3 9) Invasion de batraciens** (C) Entraînement

   (nombre de crapauds multipliés par 2 dans une boucle)

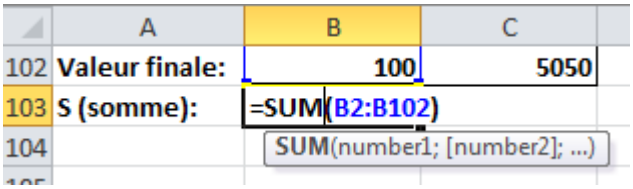
8. **Idle Somme des 101 1ers entiers** (C) Découverte

   - ✓ Calcule la somme des 100 1ers entiers naturels.
   - ✓ Réalise cet exercice d'abord sous libre office dans une feuille de calcul:

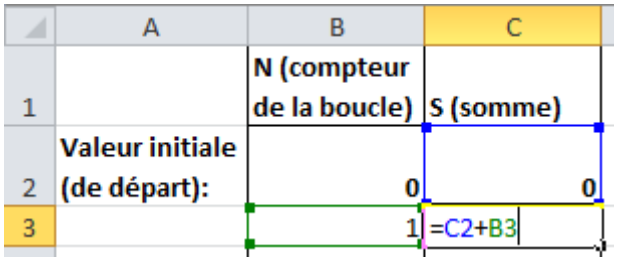
     Ecris sur une 1ère colonne « B » le compteur N de 0 à 100 grâce à la formule suivante en « B3 » puis en étirant / copiant cette formule sur les cases dessous:



     Ecris ensuite sous la colonne « B » la somme S des N entiers:



     Ecris enfin sur la colonne « C » la somme S des N entiers pour chaque ligne puis en étirant / copiant cette formule sur les cases dessous:



	A	B	C
1		N (compteur de la boucle)	S (somme)
2	Valeur initiale (de départ):	0	0
3		1	=C2+B3

Sortie

	A	B	C	D
1		N (compteur de la boucle)	S (somme)	
2	Valeur initiale (de départ):	0	0	
3		1	1	
4		2	3	
5		3	6	

...

	A	B	C	D
101		99	4950	
102	Valeur finale:	100	5050	
103	S (somme):	5050		
104				

- ✓ Réalise enfin cet exercice en programmation et pour bien comprendre le mécanisme, affiche à chaque passage dans la boucle la somme **S** des **N** entiers.

  Pour cela, tu auras besoin:

  - d'une variable N représentant la colonne « B » du compteur
  - d'une variable S représentant la colonne « C » du compteur.

  Pour ces 2 variables, repère leurs valeurs initiales, finales et le nombre de passages dans la boucle.
  En t'aidant de la feuille de calcul, écris enfin les 2 formules pour calculer N et S à chaque passage dans la boucle.

Sortie

```
N = 0      S = 0
N = 1      S = 1
N = 2      S = 3
N = 3      S = 6
....
N = 99     S = 4950
N = 100    S = 5050
```

9. Idle Nombres pairs, puissance et somme (C) Validation

a) Calcule et affiche les 50 premiers nombres **pairs:**

- ✓ soit en comptant de 2 en 2
- ✓ soit en comptant jusqu'à 50 et en affichant le double du compteur

b) Calcule et affiche les nombres impairs inférieurs à 125

c) Calcule et affiche les 30 premiers nombres **élevés au carré** ainsi que leur **somme**

d) Calcule et affiche la **somme** des 50 premiers **nombres impairs élevés au cube**

10. Fce-ioi 1.3 10) Kermesse (M) Entraînement

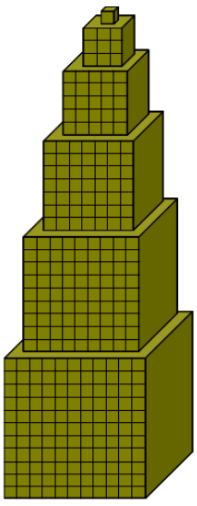
(somme dans une boucle)

11. Fce-ioi Mettre à profit les identifiants (C) Cours

12. Fce-ioi 1.3 11) Course avec les enfants (C) Validation

(ramasser et rapporter les anneaux – assez difficile !)

13. Fce-ioi 1.3 12) Construction d'une pyramide (M) Validation



14. Fce-ioi 1.3 13) Tables de multiplication (M) Challenge

15. Fce-ioi Suivre l'évolution des var. d'un prog. (C) Cours

16. Linux - Clôture de fil électrique (C) Validation

Un agriculteur veut calculer la longueur de fil électrique et le prix des piquets dont il a besoin pour clôturer un pré.
Pour cela, il fait le tour du pré en tracteur et chronomètre le temps qu'il roule à 3km/h et le temps qu'il roule à 15km/h selon la configuration du terrain.

Au total, il a conduit pendant 10 minutes à 15km/h et 30 minutes à 30km/h.

Il achète 250 piquets au prix de 3 euros chacun.

Ce que doit faire ton programme:

- ✓ Ton programme calcule et affiche la distance parcourue à 15km/h (en mètres).
- ✓ Il calcule et affiche la distance parcourue à 3km/h (en mètres).
- ✓ Il calcule et affiche la longueur de fil électrique (en mètres).
- ✓ Il calcule et affiche la distance entre 2 piquets et le prix des tous les piquets

17. Idle Bondons pour tout le monde (bis) (M) Entraînement

Rependre l'énoncé de Fce-ioi 1.3 3) Bondons pour tout le monde.
A savoir, les classes sont constituées de 25, 30, 27 et 22 élèves.

- ✓ Ton programme calcule et affiche le nombre total de bonbons lorsqu'il y a entre 0 et 12 absents sachant chaque élèves reçoit 3 bonbons (par pas d'1 élève).
- ✓ Ton programme calcule et affiche les nombre total de bonbons lorsqu'il chaque élève reçoit entre 2 et 20 bonbons sachant qu'il y a 3 absents (par pas de 2 bonbons).

18. Idle Entraînement sur les boucles (1) (C) Entraînement

- ✓ Affiche une suite de 10 nombres dont on ajoute chaque fois un nombre de plus (on ajoute 1, puis 2, puis 3, puis 4,….)

Sortie

```
1 3 6 10 15 21 28 36 45 55
```

- ✓ Affiche une suite de nombres dont un nombre est le double du précédent + 1, puis le double + 2, puis le double + 3,….

Sortie

```
1 3 8 19 42 89 184 375 758 1525
```

19. Idle Boucle s (2) Suites d'* (C) Entraînement

- ✓ Dessine la figure suivante

Astuce: tu peux (dois) utiliser une variable pour le nombre de répétitions d'une boucle. Commence par ne faire que la moitié de la figure

Sortie

```
*
**
***
****
*****
******
*******
********
*********
**********
**********
*********
********
*******
******
*****
****
***
**
*
```

20. Idle Boucle s (3) Tables de multiplication (C) Entraînement

✓ Construis la table de multiplication comme indiqué ci-dessous

Sortie

```
1 2 3 4 5 6 7 8 9 10
2 4 6 8 10 12 14 16 18 20
3 6 9 12 15 18 21 24 27 30
4 8 12 16 20 24 28 32 36 40
5 10 15 20 25 30 35 40 45 50
6 12 18 24 30 36 42 48 54 60
7 14 21 28 35 42 49 56 63 70
8 16 24 32 40 48 56 64 72 80
9 18 27 36 45 54 63 72 81 90
10 20 30 40 50 60 70 80 90 100
```

21. Idle Boucle (4) Suites de nombres (C) Entraînement

Construis la suite de nombre comme indiqué ci-dessous. Réfléchis bien: une seule variable est suffisante

Sortie

```
1
2 2
3 3 3
4 4 4 4
5 5 5 5 5
6 6 6 6 6 6
7 7 7 7 7 7 7
8 8 8 8 8 8 8 8
9 9 9 9 9 9 9 9 9
10 10 10 10 10 10 10 10 10 10
```

✓ Construis la suite de nombre comme indiqué ci-dessous. Ici tu as besoin de 2 variables

Sortie

```
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
1 2 3 4 5 6
1 2 3 4 5 6 7
1 2 3 4 5 6 7 8
1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8 9 10
```

22. Idle Boucles (5) L'horloge (M) Entraînement

✓ Affiche l'heure d'une journée sur 24 lignes, toutes les 5 minutes comme indiqué ci-dessous

Sortie

```
0 :  0 5 10 15 20 25 30 35 40 45 50 55
1 :  0 5 10 15 20 25 30 35 40 45 50 55
2 :  0 5 10 15 20 25 30 35 40 45 50 55
3 :  0 5 10 15 20 25 30 35 40 45 50 55
4 :  0 5 10 15 20 25 30 35 40 45 50 55
5 :  0 5 10 15 20 25 30 35 40 45 50 55
6 :  0 5 10 15 20 25 30 35 40 45 50 55
7 :  0 5 10 15 20 25 30 35 40 45 50 55
8 :  0 5 10 15 20 25 30 35 40 45 50 55
9 :  0 5 10 15 20 25 30 35 40 45 50 55
10 :  0 5 10 15 20 25 30 35 40 45 50 55
11 :  0 5 10 15 20 25 30 35 40 45 50 55
12 :  0 5 10 15 20 25 30 35 40 45 50 55
13 :  0 5 10 15 20 25 30 35 40 45 50 55
14 :  0 5 10 15 20 25 30 35 40 45 50 55
15 :  0 5 10 15 20 25 30 35 40 45 50 55
16 :  0 5 10 15 20 25 30 35 40 45 50 55
17 :  0 5 10 15 20 25 30 35 40 45 50 55
18 :  0 5 10 15 20 25 30 35 40 45 50 55
19 :  0 5 10 15 20 25 30 35 40 45 50 55
20 :  0 5 10 15 20 25 30 35 40 45 50 55
21 :  0 5 10 15 20 25 30 35 40 45 50 55
22 :  0 5 10 15 20 25 30 35 40 45 50 55
23 :  0 5 10 15 20 25 30 35 40 45 50 55
```

23. Idle Boucle s (6) Pyramide (M) Entraînement

Affiche, sur 10 lignes, une suite de nombres consécutifs. Chaque ligne commence par le nombre supérieur par rapport à la ligne précédente

Sortie

```
1 2 3 4 5 6 7 8 9 10
2 3 4 5 6 7 8 9 10 11
3 4 5 6 7 8 9 10 11 12
4 5 6 7 8 9 10 11 12 13
5 6 7 8 9 10 11 12 13 14
6 7 8 9 10 11 12 13 14 15
7 8 9 10 11 12 13 14 15 16
8 9 10 11 12 13 14 15 16 17
9 10 11 12 13 14 15 16 17 18
10 11 12 13 14 15 16 17 18 19
```

24. Idle – Robot synthèse tp3

- ✓ Télécharge du drive le dossier **tp3_algorea_synthese**, décompresse les fichiers et déplace-les dans ton dossier de travail.
- ✓ Modifie et exécute le fichier **test.py**. L'appel des test1, 2, 3, 4 se fait en modifiant l'instruction en fin de code test1() par test2(),test3(), test4()

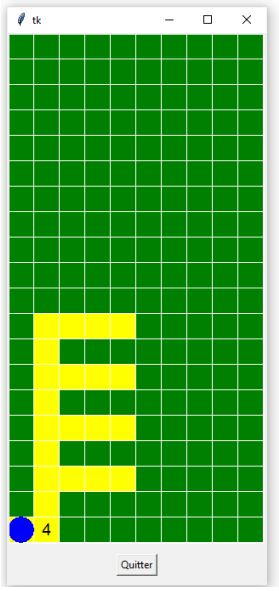
  Attention tu dois écrire une seule fonction test() qui est appelé par test1() par test2(),test3().

  test4() est un test séparé
- ✓ Déplace le point bleu en passant par toutes les cases jaunes.

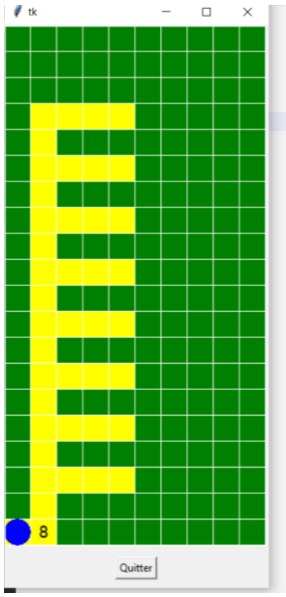
  Tu dois utiliser un minimum d'instructions, le déplacement se fait grâce à :

  est(), ouest(), nord(), sud(), nombreSurCase()
- ✓ Démarche :
  - Pense à bien repérer sur la figure le bout de dessin qui se répète.
  - Ecris tout d'abord le code pour parcourir ce bout de dessin.
  - Complète-le pour parcourir toute la figure.

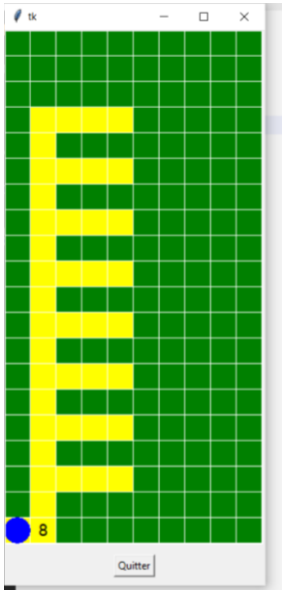
test1



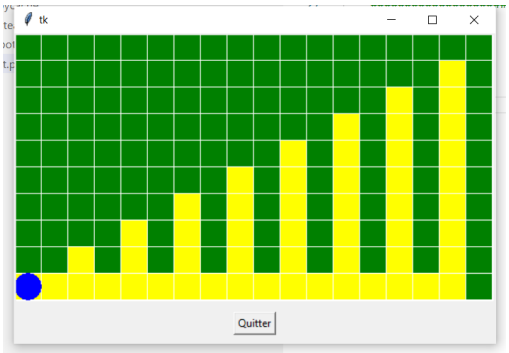
test2



test3



test4



25. Idle – Robot challenge tp3

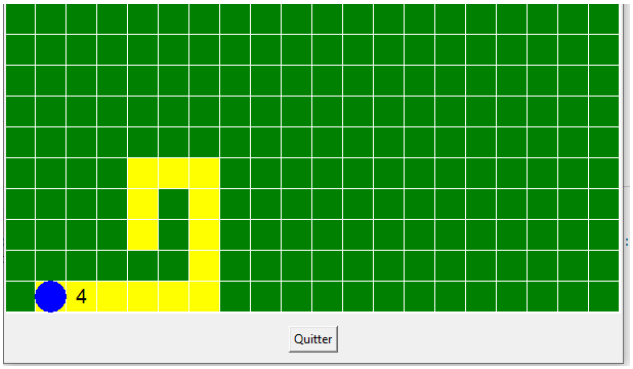
- ✓ Télécharge du drive le dossier **tp3_algorea_chalenge**, décompresse les fichiers et déplace-les dans ton dossier de travail.
- ✓ Modifie et exécute le fichier **test.py**. L'appel des test1, 2 se fait en modifiant l'instruction en fin de code test1() par test2()

  Attention tu dois écrire une seule fonction test() qui est appelé par test1() par test2().
- ✓ Déplace le point bleu en passant par toutes les cases jaunes.

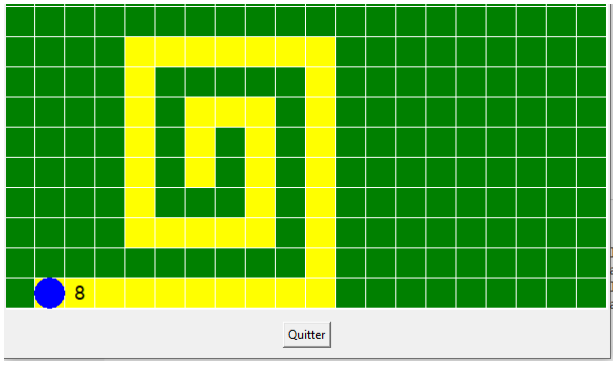
  Tu dois utiliser un minimum d'instructions, le déplacement se fait grâce à :

  avance(), droite(), gauche(), nombreSurCase()

test1



test2



4 TP4 Fce-ioi-4 : Lecture de l'entrée

Consignes (rappel)

A la fin de chaque exercice:

- Sur Fce-ioi ou algorea: copie ton code dans un fichier ***tp4_titre-de-l'-exerice.py***
- Livre tes fichiers sur le drive dans la rubrique:
  zone_transfert_eleves_Info4TTi -> xx_ton-prenom -> 20aa/mm/jj -> tp4

Synthèse de cours

⇨ **Lis en complément le petit cours suivant :**

https://e-nsi.forge.aeif.fr/init_python/4-input/23-lecture

1. Fce-ioi Des programmes interactifs (C) Cours

⇨ A lire pour bien comprendre la différence entre une entrée et une sortie.

2. Idle Test1: Quel âge as-tu ? (C) Découverte

✓ Ouvre un terminal, crée un dossier tp4, crée un fichier xx_tp3-test1_age.c

Ce que doit faire ton programme

Ton programme doit lire un entier qui correspond à ton Age et afficher ton année de naissance et ton âge en 2050.

Entrées

```
17
```

Sorties

```
2002
50
```

Pour que ton programme soit compréhensible, il faut évidemment rajouter des instructions pour préciser à l'utilisateur ce qu'il doit rentrer et ce que calcule le programme.

Sortie

Le résultat final attendu sera ainsi:

```
Quel age as-tu ?
17
Ah ! Tu as donc 17 ans !
Tu es donc ne(e) en 2000 !
Et en 2050, tu auras 50 ans !
```

- ✓ La lecture d'une variable au clavier se fait avec l'instruction **input()**. Cette variable est par défaut de type text et peut être convertie en entier par l'instruction **int()** ou en réel **par float()**.

Pour lire ton âge il faut écrire:

```
age=input()
age=int(age)
```

Ou plus simplement:

```
age=int(input())
```

3. Fce-ioi 1.4.2) test2: Retraite spirituelle (M) Entraînement

Une fois par an, tout habitant d'un village de plus de 15 ans doit effectuer une randonnée spirituelle, celle-ci pouvant durer plusieurs jours. La durée dépendra du temps nécessaire à chaque personne pour faire le bilan de l'année écoulée. Au cours de cette randonnée, la personne doit répéter encore et encore la même incantation, une fois par minute.
Tu te demandes combien de fois au total l'incantation aura été répétée, selon la durée totale de la randonnée.

Ce que doit faire ton programme

Ton programme devra lire un entier, le nombre de jours que dure la randonnée. Ensuite, il devra afficher le nombre de fois que l'incantation est répétée sachant qu'elle est dite une fois par minute, 16 heures par jour (les 8 autres heures, on dort !).

Entrées

```
30
```

Sorties

```
28800
```

Sortie

```
Entre le nombre de jours que dure la randonnée:
30
L'incantation est répétée 28800 fois pendant la randonnée
```

4. Idle test3: Champ de Moutab (C) Entraînement

Des villageois font pousser dans leurs champs carrés un étrange légume géant: le Moutab. Les anciens vous ont dit que, d'après leurs estimations, le rendement de cette année serait de 23 kg de Moutab par mètre carré de culture. Vous voudriez écrire un programme permettant aux villageois de prédire la quantité de légume qu'ils peuvent espérer récolter dans chacun de leurs champs.

Ce que doit faire ton programme

Ton programme doit lire deux entiers qui représentent la longueur et la largeur d'un champ rectangulaire. Il doit ensuite afficher la masse que l'on pourra récolter de ce champ si l'on suppose que la production sera de 23 kg par mètre carré.

Entrées

```
10
20
```

Sorties

```
4600
```

Pour que ton programme soit compréhensible, il faut évidemment rajouter des instructions pour préciser à l'utilisateur ce qu'il doit rentrer et ce que calcule le programme.

Sortie

```
A toi à l'écrire !
```

Ce que doit faire ton programme

Modifie ton programme pour faire le calcul quel que soit le rendement tapé au clavier par l'utilisateur.
Modifie ton programme pour calculer la quantité de légume par habitant ; le nombre d'habitants qui vont se partager le Moutab étant entré par l'utilisateur.

5. Fce-ioi 1.4.5) test4: Graduation de thermomètre (C) Entraînement

Les habitants du village utilisent beaucoup de thermomètres différents: certains pour l'été, d'autres pour l'hiver, d'autres pour la température de l'eau, etc. Ce qui change d'un thermomètre à l'autre, ce sont les valeurs de la température minimale et de la température maximale lisibles sur la graduation. Les habitants aimeraient pouvoir imprimer facilement la graduation des températures possibles pour chaque thermomètre.

Ce que doit faire ton programme

Étant données deux températures entières *tempMin* et *tempMax*, votre programme doit afficher toutes les températures comprises entre les deux, bornes incluses.

Entrées

```
9
14
```

Sorties

```
9
10
11
```

```
12
13
14
```

6. Fce-ioi Utiliser les exemples des sujets (C) Cours

7. Fce-ioi 1.4.6) test5: Jeu de calcul mental (C) Validation

Au village, la passion pour le calcul mental est une tradition: des jeux centrés sur cette pratique sont régulièrement organisés par les habitants. Pour chaque jeu, ils décident d'abord combien de nombres devront être prononcés ; puis chaque joueur doit effectuer un calcul déterminé par les règles du jeu. Chaque fois que quelqu'un se trompe et qu'un autre joueur s'en rend compte, le joueur qui s'est trompé doit se corriger, et il devra un Gombo (une friandise du coin) à celui qui lui a signalé son erreur le plus rapidement.
Vous aimeriez participer, mais les habitants vont si vite et manipulent des nombres si grands que vous êtes tout de suite dépassé par les calculs ! Alors qu'un nouveau jeu se prépare, vous décidez finalement d'utiliser votre robot pour vous aider à rivaliser.

Ce que doit faire ton programme

Un nombre de départ va être donné par le chef du village. La personne qui suit doit le multiplier par 2, puis la suivante doit multiplier le nombre obtenu par 3, celle d'encore après doit multiplier le résultat par 4… jusqu'à ce que les *nbNombres* calculs aient été effectués.
Le chef a choisi le nombre 66 pour démarrer le jeu. Votre programme lira la quantité de nombres attendue par le jeu (nombre de départ inclus). Il devra ensuite afficher tous les nombres de la partie afin de vous rendre imbattable !

Entrées

```
4
```

Sorties

```
66
132
396
1584
```

8. Fce-ioi 1.4.3) test6: Age des petits enfants (M) Entraînement
9. Fce-ioi 1.4.4) test7: Encore des punitions (M) Entraînement
10. Fce-ioi 1.4.7) test8: Grande braderie (C) Validation
11. Fce-ioi 1.4.8) test9: Bétail (M) Entraînement
12. Fce-ioi 1.4.9) test10: Socles pour statues (C) Validation
13. Fce-ioi Instructions condensées (C) Cours
14. Fce-ioi 1.4.9) test11: Le plus beau Karvas (C) Validation
15. Fce-ioi Erreur si on lit trop de choses (C) Cours
16. Fce-ioi Portée d'une variables (C) Cours

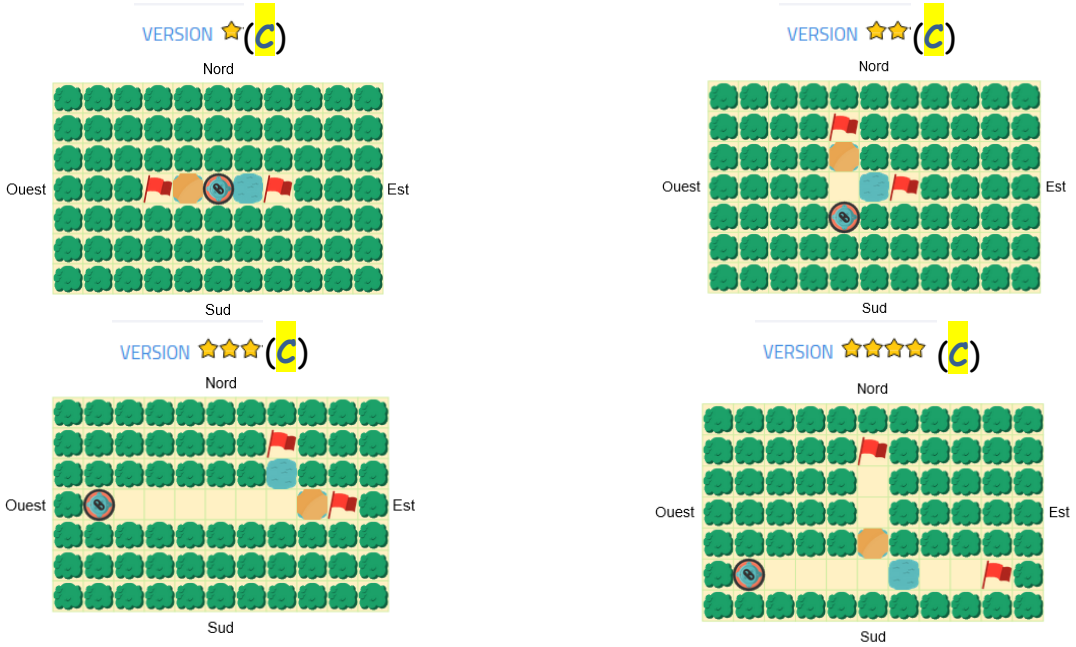
5 TP5 – Tests et conditions

Consignes (rappel) : à la fin de chaque exercice:

- Sur Fce-ioi ou algorea: copie ton code dans un fichier ***tp5_titre-exercice.py***
- Livre tes fichiers sur le drive dans la rubrique:
  zone_transfert_eleves_Info4TTi -> prenom -> 20aa/mm/jj -> tp5

5.1 Algorea-5 : Faire des choix en fonction des éléments de la grille

1. Algo – Trouver la sortie



1. Algo – Planter des fleurs

5.2 Fce-ioi-5 : Tests et conditions

1. Fce-ioi – 1.5.1) test1: transport de bagage (C) Découverte
2. Idle test2: quel âge as-tu ? (C) Découverte

Ce que doit faire ton programme

Ton programme doit lire un entier qui correspond à ton Age.
Si ton âge est strictement supérieur à 18 ans, il affiche « Tu es majeur ! ».
Si ton âge est égal à 18 ans, il affiche « Tu es juste majeur »
Si ton âge est inférieur ou égal à 15 ans, il affiche « Tu es encore un gamin ! »
Si ton âge est compris entre 15 et 18 ans, il affiche « Tu es un ado ! »

Exemple1 - Entrées

```
19
```

Sorties

```
Tu es majeur !
```

Exemple2 - Entrées

```
15
```

Sorties

```
Tu es encore un gamin !
```

Exemple3 - Entrées

```
17
```

Sorties

```
Tu es un ado !
```

Exemple4 - Entrées

```
18
```

Sorties

```
Tu es juste majeur
```

Code python

- ✓ Les instructions de test que tu peux utiliser sont:

```
if (condition1):
    instruction1
else if (condition2):
    instruction2
else:
    instruction3
```

- ⇨ On le traduira en Français par:
  Si la condition1 est vérifiée, le programme exécutera la condition1,
  Sinon, si la condition2 est vérifiée, le programme exécutera la condition2,
  Sinon (sous-entendu, si aucune des autres conditions n'est vérifiée), le programme exécutera la condition3

- ✓ Les comparateurs pour tester les conditions sont:

Comparateur	Signification
==	est égal à
>	est strictement supérieur à
<	est strictement inférieur à
>=	est supérieur ou égal à
<=	est inférieur ou égal à
!=	est différent de

```
age = int(input ("Quel age as-tu?"))
if age > 18:
   print("Tu es majeur ! ")
elif age == 18:
   print("Tu es juste majeur !")
elif age <= 15:
   print("Tu es encore un gamin ! ")
else:                                   # sous-entendu tu as entre 15 et 18 ans
   print("Tu es un ado ! ")
```

Remarques:

- Si tu veux exécuter plusieurs instructions après le test, n'oublie pas de les **tabuler**
- Regarde bien où sont mis les «: »

Synthèse de cours

- ⇨ **Lis en complément le petit cours suivant :**
  https://e-nsi.forge.aeif.fr/init_python/5-if/26-conditions

3. Idle test3: Champ de Moutab (C) Entraînement

Modifie le programme du tp4-test3 sur le champ de Moutab pour afficher:

- « c'est une bonne récolte » si la quantité de Moutab par habitant est supérieure à 200 000 kg
- « c'est une mauvais récolte » si la quantité de Moutab par habitant est supérieure à 100 000 kg
- « c'est une récolte standard » si la quantité de Moutab par habitant est comprise entre 100 000 et 200 000 kg

4. Fce-ioi 1.5.3) test4: Tarifs dégressifs (C) Entraînement

L'auberge dans laquelle vous avez prévu de passer la nuit ce soir propose des tarifs très intéressants, pour peu que l'on n'arrive pas trop tard. En effet, plus on arrive tôt moins on devra payer. Vous essayez de construire un programme vous donnant directement le prix à payer en fonction de votre heure d'arrivée.

Ce que doit faire ton programme

Ton programme lira un entier, l'heure d'arrivée, qui sera compris entre 0 et 12 inclus. 0 correspond à midi, 1 à 1h de l'après-midi, etc. et 12 à minuit.
Le prix de la chambre est de 10 pièces à midi, et augmente de 5 pièces chaque heure après midi. Il est donc de 15 pièces à 13h, etc. Il ne peut cependant pas dépasser 53 pièces.
Ton programme devra afficher le prix à payer correspondant à l'heure d'arrivée donnée.

Exemple1 - Entrées

```
7
```

Sorties

```
45
```

Exemple2 - Entrées

```
10
```

Sorties

```
53
```

5. **Fce-ioi Conditions: erreur possible** (C) Cours

6. **Fce-ioi Blocs formés de plusieurs instructions** (C) Cours

7. **Fce-ioi 1.5.6) test5: Traversée du pont** (M) Entraînement

8. **Fce-ioi 1.5.7) test6: Concours de tir à la corde** (C) Validation

Vous avez passé la nuit dans une auberge. Au petit matin, un championnat de tir à la corde y est organisé. Vous ne souhaitez pas participer, mais l'aubergiste insiste pour que vous soyez impliqué dans l'événement. Vous décidez alors de vous engager dans les paris qui se font sur les deux équipes qui concourent.
Le championnat oppose deux équipes, contenant chacune autant de joueurs. Pour donner de l'allure et pimenter les paris, au début du championnat, tous les joueurs sont présentés, avec leur poids, avant d'aller tenir leur côté de la corde. Il est d'abord présenté un membre de la première équipe, puis de la deuxième, puis de la première, puis de la deuxième etc. jusqu'à ce que tous les joueurs soient passés. Afin de vous faire un premier pronostic, vous calculez le poids total de chaque équipe, avec votre robot.

Ce que doit faire ton programme

Ton programme devra lire un premier entier: le nombre de membres qui constituent une équipe.
Ensuite, il devra lire les poids des joueurs (en kilogrammes), sachant que le 1er poids est celui d'un joueur de la 1ère équipe, le 2ème poids celui d'un joueur de la 2ème équipe, le 3ème poids la 1ère équipe, le 4ème poids la 2ème équipe, etc.
Après avoir calculé le poids total de chaque équipe, ton programme doit afficher l'équipe avantagée, cad celle qui a le plus grand poids.
(voir l'exemple ci-dessous).

On vous garantit que les deux équipes n'auront pas le même poids total.

Exemple - Entrées

```
3
40
80
50
50
60
10
```

Sorties

```
L'équipe 1 a un avantage
Poids total pour l'équipe 1: 150 kg
Poids total pour l'équipe 2: 140 kg
```

Exécution du code python

```
Combien de membres possèdent tes équipes?
3
Poids du joueur de la 1ère équipe:
40
Poids du joueur de la 2ème équipe:
80
Poids du joueur de la 1ère équipe:
50
Poids du joueur de la 2ème équipe:
50
Poids du joueur de la 1ère équipe:
60
Poids du joueur de la 2ème équipe:
10
L'équipe 1 a un avantage
Poids total pour l'équipe 1: 150 kg
Poids total pour l'équipe 2: 140 kg
```

9. Fce-ioi 1.5.8) test7: Mot de passe du village (C) Validation

6 TP6 – Fce-ioi-6 : Structures avancées

1. **Fce-ioi Structures imbriquées** (C) Cours

2. **Fce-ioi 1.6.1 test1: Villes et villages** (C) Entraînement

   Compteur conditionné

3. **Fce-ioi 1.6.2 test2: Planning de la journée** (C) Validation

   Il faut utiliser un compteur conditionné (un compteur après un if …)
   Tu peux calculer une valeur absolue avec abs() à condition d'avoir importé la librairie mathématique.

Exemple

```
from math import *
print(abs(-3))
```

Sortie

```
3
```

4. **Fce-ioi 1.6.3 test3: Etape la plus longue** (C) Découverte

   Demande de l'aide pour te faire expliquer comment rechercher un maximum

5. **Fce-ioi 1.6.4 test4: Calcul des dénivelées** (M) Entraînement

   Il faut utiliser 2 compteurs conditionnés

6. **Fce-ioi 1.6.6 test5: Tarifs de l'auberge** (M) Entraînement

7. **Fce-ioi 1.6.7 test6: Protection du village** (M) Entraînement

8. **Fce-ioi 1.6.8 test7: Le juste prix** (C) Validation

   Calcul du minimum avec la même technique que pour le calcul du maximum

7 TP7 Fce-ioi-7 : Conditions avancées, opérateurs booléens

1. **Fce-ioi 1.7.1) test1: Espion étranger** (C) Découverte

   Compteur conditionné

2. **Fce-ioi 1.7.2) test2: Maison de l'espion** (M) Entraînement

3. **Fce-ioi 1.7.3) test3: Nb de jours dans le mois** (C) Entraînement

4. **Fce-ioi 1.7.4) test4: Amitié entre gardes** (M) Entraînement

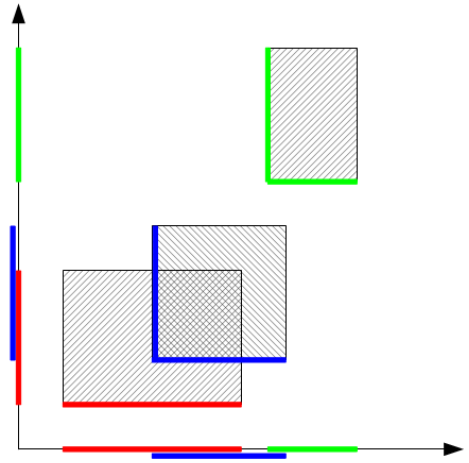
Aide à la résolution:

- ⇨ Liste avec un dessin les cas où les soldats sont amis et ceux où ils ne sont pas amis
- ⇨ Tu peux en conclure qu'il est plus simple d'écrire ton programme en testant si les soldats ne sont PAS amis

5. **Fce-ioi 1.7.6) test5: Caserne de pompier** (M) Challenge

Aide à la résolution:

Le point principal de l'algorithme consiste à tester l'intersection de deux rectangles. En regardant sur un dessin on voit que pour tester une intersection il faut regarder la "projection" du rectangle sur les deux axes. Si les segments correspondants s'intersectent sur chaque axe alors les rectangles s'intersectent, sinon les rectangles sont disjoints.



Les rectangles ne s'intersectent donc pas si au moins un des couples de segments ne s'intersecte pas. Le programme est alors relativement simple.

6. **Fce-ioi 1.7.7) test6: Personne disparue** (C) Découverte

   Flag

7. **Fce-ioi 1.7.8) test7: La grande fête** (M) Entraînement

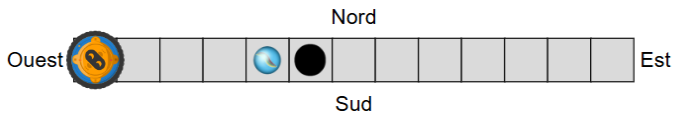
   Opérateurs booléens

8. **Fce-ioi 1.7.10) test8: Zones de couleurs** (M) Challenge

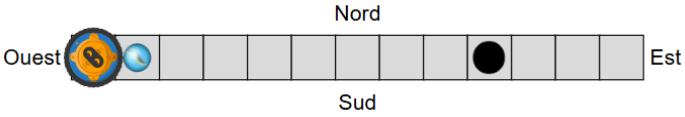
8 TP8 – Répétitions conditionnées (boucle while)

8.1 Algorea-7 : Donner la même séquence au robot jusqu'à qu'une condition soit remplie

1. Ranger les billes

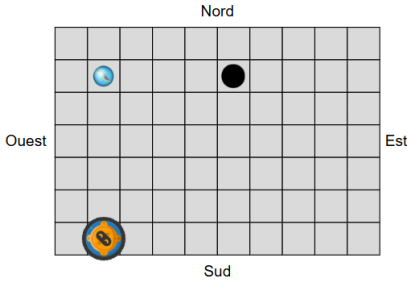


VERSION ★ (C)

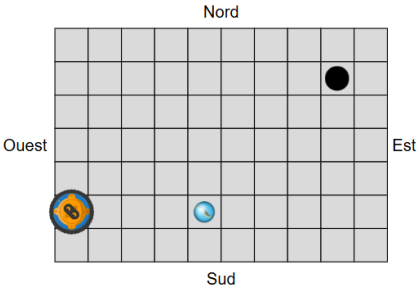


VERSION ★★ (C)

VERSION ★★★ (C)

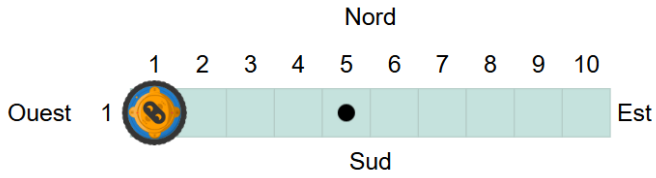


VERSION ★★★★ (M)

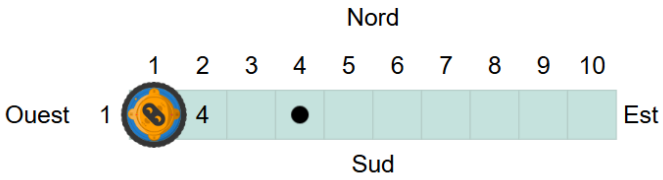


2. Peindre le motif

VERSION ★ (C) VERSION ★★ (C) VERSION ★★★ (C) VERSION ★★★★ (M)

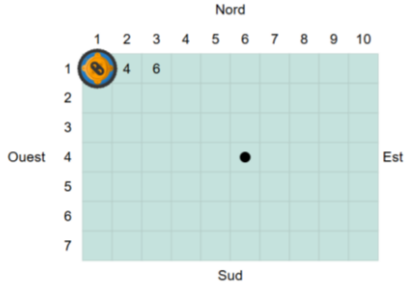


VERSION ★ (C)

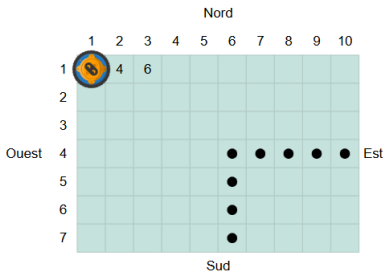


VERSION ★★ (C)

VERSION ★★★ (C)



VERSION ★★★★ (M)



8.2 Fce-ioi-2 : Répétitions conditionnées

1. **Fce-ioi 1.8.1) test1: Contrôle d'une épidémie** (C) Découverte

   ⇨ Une même personne en contamine 2 TOUS les jours.
   Ne pas oublier de compter celle qui contamine dans le décompte !

2. **Fce-ioi Boucle infinie** (C) Cours

3. **Fce-ioi 1.8.2) test2: Comptes annuels** (M) Entraînement

4. **Fce-ioi 1.8.3) test3: C'est + / c'est -** (M) Entraînement

Il plus intéressant de modifier l'énoncé de France-ioi en faisant tirer par le programme un nombre au hazard.
Tirer un entier entre 0 et 100 se fait avec la fonction **randint**.
Il faut l'importer de la librairie **random**.
Teste cette fonction avec le code suivant (**CTRL+C** pour arrêter le programme)

Code

```
from random import *
while(1):
    print(randint(0,100))
```

5. **Fce-ioi 1.8.4) test4: Pyramide** (M) Challenge

Aide à la résolution:

La principale difficulté de cet exercice est de trouver la bonne condition pour la boucle « tant que ». On veut en effet construire la plus grande pyramide avec moins de pierres que la limite donnée.
Pose-toi la question suivante: est-ce que le nombre de pierres de cette pyramide, si je lui ajoutais un étage, est plus petit que le nombre maximum de pierres autorisées ? Si oui, alors on peut en toute sécurité ajouter un étage ; sinon, on sait qu'il faut s'arrêter, et le nombre de pierres de la pyramide actuelle est la réponse attendue.

6. **Fce-ioi 1.8.5) test5: Mélange explosif** (M) Validation

9 TP9 – Future coder : chaînes, une alternative à FceIOI

Va sur le site de [11] Tutoriel en ligne **furture coder**
https://fr.futurecoder.io/

Le site est récent, très bien fait, les explications sont claires.
Par contre, les exercices d'entraînement sont peu nombreux et vont peut-être devenir vite difficile.

- ✓ Inscris-toi afin que le site se souvienne de ta progression
- ✓ Une fois connecté, dans les réglages, tu peux activer le mode « **développeur** » qui permet de passer une étape si tu es bloqué.
- ✓ Attention, tes exercices ne sont pas sauvegardés, mieux vaut travailler avec un fichier ouvert sur ton pc et faire des copier/coller des exercices résolus au fur et à mesure dans un fichier sous idle.
- ✓ En cas de difficulté, tu peux débugger ton code en l'exécutant pas à pas avec : Python Tutor

Les chapitres ci-dessous reprennent ceux de FutureCoder que tu dois faire ne classe.
Trouve un exemple de code pour chaque instruction surlignée en jaune, et copie-le dans ton memo.

9.1 FutureCoder: La console et bases sur les chaînes (C) Découverte

+ avec des chaînes de caractères : la **concaténation**

9.2 FutureCoder: Variables (chaînes) (C) Découverte

Concaténation de variables de type chaînes de caractère

9.3 FutureCoder : Boucles for (chaînes)

1. FC - Présentation des boucles for (C) Découverte

for … in : …

2. FC - Indentation (C) Découverte

Indentation = tabulation = 4 espaces

3. FC - Exercices simples sur les boucles for (C) Entraînement

```
---W
---o
---r
---l
---d
```

```
World
World
World
World
World
```

4. FC - Construction de chaînes (C) Découverte

```
Hello
Hello!
```

```
-W
-Wo
-Wor
-Worl
-World
```

```
-World
-WorldW
-WorldWo
-WorldWor
-WorldWorl
```

5. FC - Exercices de construction de chaînes (C) Entraînement

Partie 1 (C) Entraînement

```
-W
-W o
-W o r
-W o r l
-W o r l d
```

Partie 2 (C) Entraînement

```
W
Wo
Wor
Worl
World
```

Partie 2 (C) Entraînement

```
W
oW
roW
lroW
dlroW
```

Partie 4 (C) Entraînement

```
+-----+
|World|
+-----+
```

Partie 5 (M) Challenge

```
+World+
W     W
o     o
r     r
l     l
d     d
+World+
```

Partie 6 (M) Challenge

```
W
 o
  r
   l
    d
```

10 TP10 – Les listes

10.1 FutureCoder : une alternative à FceIOI

1. FC - Présentation des listes (C) Découverte

for … in : …
Addition des éléments d'une liste

Bonus facultatif (M) Challenge

```
Ceci - est - une - liste
```

2. FC - Construction de nouvelles listes (C) Découverte

Rajout d'un élément (**+[…]** ou **append(…)**)

3. FC - Utilisation du break (C) Découverte

break
if … in … : "for chose in choses: if chose == cible: … " peut être remplacée par "if chose in choses: …"

4. FC - Lecture d'éléments à une position, range() et len() (C) Découverte

for i in range(…), len()

Exercices élémentaires (M) Entraînement

5. FC - Exercices avec range et len() (C) Entraînement

Trouver l'indice d'un élément (C) Entraînement

1. Exercices parcours de 2 chaînes par leur indice (C) Entraînement

```
H W
e o
l r
l l
o d
```

2. Exercices parcours de 2 chaînes de tailles différentes (M) Challenge

10.2 FC - Vocabulaires sur les appels de fonctions (M) Découverte

10.3 FC – Fonctions et méthodes sur les listes (C) Découverte

append , len, range, [] (C) Découverte

index, pop, remove (C) Découverte

Exercices (M) Entraînement

10.4 Linux - Exercices complémentaires sur les listes

1. Linux test1: Mois de l'année (C) Entraînement

Sois la liste des mois de l'année :

```
mois_annee =
['Janvier','Février','Mars','Avril','Mai','Juin','Juillet','Août','Septembre','Octobre','
Novembre','Décembre']
```

1. Affiche tous les mois sur une ligne séparés par un espace

Sortie

```
Janvier Février Mars Avril Mai Juin Juillet Août Septembre Octobre Novembre Décembre
```

2. Affiche chaque mois ainsi que sa position sur une ligne

Sortie

```
0 Janvier
1 Février
2 Mars
3 Avril
4 Mai
5 Juin
6 Juillet
7 Août
8 Septembre
9 Octobre
10 Novembre
11 Décembre
```

3. Affiche le nom du mois dont la position est entrée par l'utilisateur

Entrée

```
8
```

Sortie

```
Septembre
```

4. Modifie un mois de la liste, la position et son nom étant entrés par l'utilisateur

Entrée

```
3

Germinal
```

Sortie

```
['Janvier','Février','Mars',Germinal,'Mai','Juin','Juillet','Août','Septembre','Octobre',
'Novembre','Décembre']
```

5. Affiche si un mois entré par l'utilisateur est présent ou pas dans la liste.

Entrée

```
Avril
```

Sortie

```
N'est pas dans la liste
```

Entrée

```
Janvier
```

Sortie

```
Est dans la liste
```

6. **Rajoute un mois entré par l'utilisateur en fin de liste**

Entrée

```
13emois
```

Sortie

```
['Janvier','Février','Mars',Avril,'Mai','Juin','Juillet','Août','Septembre','Octobre','No
vembre','Décembre','13emois']
```

2. Linux test2: Nombre de jours des mois (C) Entraînement

Sois la liste du nombre de jours des mois de l'année :

```
nb_jours= [31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31]
```

1. **Crée une liste contenant les mois de l'année ainsi que leur nombre de jours**

Sortie

```
['Janvier', 31, 'Février', 28, 'Mars', 31, 'Avril', 30, 'Mai', 31, 'Juin', 30, 'Juillet',
31, 'Août', 31, 'Septembre', 30, 'Octobre', 31, 'Novembre', 30, 'Décembre', 31]
```

2. **Calcule le nombre de jours moyen d'un mois de l'année**

Sortie

```
30.5
```

3. **Affiche le nombre de jours minimum dans un mois**

Sortie

```
29
```

4. **Affiche le mois qui a le moins de jours**

Sortie

```
Février
```

5. **Calcule le nombre de mois qui ont 31 jours**

Sortie

```
7
```

6. **Affiche les mois qui ont plus de 31 jours**

Sortie

```
['Janvier','Mars','Mai','Juillet','Août','Octobre','Décembre']
```

France IOI niveau 2

11 TP11 - FceIOI niveau2.2: Découverte des tableaux ou listes

1. Fce-ioi La boucle for (C) Cours

   ⇨ **Lis le cours de FceIOI tout en testant le code proposé sous linux.**

2. FutureCoder : Boucles for (C) Cours

Suis les chapitres suivants tout en sauvegardant ton code avec idle :

1. **Présentation des boucles for**
2. **Indentation**
3. **Exercices simples sur les boucles for**

3. FutureCoder : Listes (C) Cours

Suis les chapitres suivants tout en sauvegardant ton code avec idle :

1. **Présentation des listes**
4. **Construction de nouvelles listes**
5. **Utilisation de break pour sortir prématurément d'une boucle**
6. **Lecture d'éléments à une position, range() et len()**

4. Fce-ioi 2.2.1) test1: Préparation de l'onguent (C) Validation

   ⇨ **Avant de faire l'exercice, lis le cours de FceIOI tout en testant le code proposé sous linux.**

5. Fce-ioi Accès en dehors du tableau (C) Cours

6. Linux test2: Manipuler les listes (C) Entraînement

   1. Reprends l'exemple de Fce-ioi dans le cours sur les mois de l'année et affiche sur une même ligne le contenu du tableau pour chaque position (on parlera d'index au lieu de position d'où l'utilisation de la variable i):

nbJours											
0	1	2	3	4	5	6	7	8	9	10	11
31	29	31	30	31	30	31	31	30	31	30	31

Code:

```
nbJours = [31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31]
print("i   nbJours")
for i in range(12):
    print(i,"   ", nbJours)
```

Sortie:

```
i       nbJours
0       31
1       29
2       31
3       30
4       31
5       30
6       31
7       31
8       30
9       31
10  30
11  31
```

7. affiche le nombre de jours du mois n° 2.

8. affiche le nombre de jours du n° du mois demandé par l'utilisateur.

Exemple d'entrée:

```
10
```

Sortie:

```
30
```

9. demande à l'utilisateur le n° du mois qu'il veut changer ainsi que le nouveau nombre de jours puis affiche la liste modifiée

Exemple d'entrée:

```
8
35
```

Sortie:

```
[31, 29, 31, 30, 31, 30, 31, 31, 35, 31, 30, 31]
```

10. demande à l'utilisateur un nombre jours, vérifie s'il est dans la liste et si c'est le cas, affiche sa(ses) position(s)

Exemple d'entrée:

```
30
```

Sortie:

```
1
3 5 8 10
```

7. Linux test3: Moyenne d'une liste (C) Entraînement

A partir d'une liste de notes de 10 élèves, calcule la somme de ces notes et déduis-en leur moyenne.

8. Linux test4: Maximum d'une liste (C) Entraînement

A partir d'une liste de notes de 10 élèves, trouve la meilleur note et affiche sa position dans la lsite

9. Fce-ioi 2.2.2) test5: Liste de courses (C) Entraînement

Les stocks des ingrédients nécessaires à la réalisation de l'onguent commencent à se vider et les savants vous chargent d'aller en ville acheter une certaine quantité de chaque ingrédient, afin de pouvoir continuer la production pendant le prochain mois.
Le comptable étant particulièrement pointilleux, il vous donnera exactement la quantité d'argent dont vous avez besoin, pas une pièce de plus. Heureusement vous savez à l'avance le prix de chaque ingrédient et la quantité dont vous avez besoin.

Ce que doit faire votre programme :

Il y a 10 ingrédients et ils ont tous un prix au kilo différent : 9, 5, 12, 15, 7, 42, 13, 10, 1 et 20.
Votre programme devra lire 10 entiers, le poids (en kilogrammes) qu'il faut acheter pour chaque ingrédient. Il devra calculer le coût total de ces achats.

Exemple d'entrée :

```
1
1
1
1
1
1
1
1
1
1
```

Sortie :

```
134
```

10. Fce-ioi 2.2.3) test6: Grand inventaire (M) Découverte
11. Fce-ioi 2.2.4) test7: Etude de marché (M) Découverte
12. Fce-ioi 2.2.5) test8: Répartition du poids (M) Entraînement
13. Fce-ioi 2.2.6) test9: Visite de la mine (M) Challenge

L'un des produits nécessaires pour la fabrication de l'onguent magique, un minerai très rare, ne se trouve qu'en un seul endroit sur la planète, au fond de la plus vieille mine existante, jadis exploitée par le peuple nain. Désormais seuls quelques uns d'entre eux sont encore sur place, afin de guider les voyageurs (commerçants et touristes) au sein de ce dédale de cavernes et galeries.
Après avoir engagé un guide, il vous mène jusqu'à l'endroit prévu mais un petit désaccord sur le paiement de ses services le pousse à vous laisser sur place, sans aucune chance de retrouver la sortie. Heureusement votre robot à conservé en mémoire la suite des déplacements qui vous ont amené de l'entrée jusqu'à votre position actuelle, il ne vous reste plus qu'à suivre le chemin inverse !
Ce que doit faire votre programme :
Il existe 5 types de déplacements, représentés par 5 entiers différents : aller à gauche (1), aller à droite (2), aller tout droit (3), monter (4) et descendre (5).
Le premier entier à lire est le nombre total de déplacements (1000 au maximum). Ensuite, chaque déplacement (représenté par un entier) est indiqué sur sa propre ligne.
Vous devez afficher la suite des déplacements à faire pour aller de votre position actuelle à la sortie.

Exemple d'entrée :

```
6
1
3
2
4
4
5
```

Sortie :

```
4
5
5
1
3
2
```

Aide à la résolution:

Dans le principe, on commence par lire tous les déplacements faits puis il faut les afficher, en partant de la fin, en inversant à chaque fois les déplacements. On pourrait donc avoir un code du type

```
Si deplacement == 1:
  deplacement = 2
SinonSi deplacement == 2:
  deplacement  = 1
SinonSi deplacement == 4:
  deplacement = 5
SinonSi deplacement= = 5:
deplacement = 4
```

C'est cependant assez répétitif et on sent qu'il est possible de faire mieux. On peut en effet utiliser un tableau qui nous donnera, pour chaque déplacement, le déplacement inverse:

```
deplacementInverse = [0, 2, 1, 3, 5, 4]
deplacement = deplacementInverse[deplacement]
```

Cette technique est très courante et permet d'avoir des codes plus courts. Vous aurez l'occasion de la réutiliser à de nombreuses occasions.

14. Fce-ioi 2.2.7) test10: Journée des cadeaux (M) Découverte
15. Fce-ioi Affichage simplifié (C) Cours
16. Fce-ioi 2.2.8) test11: Course à 3 jambes (M) Entraînement
17. Fce-ioi 2.2.9) test12: Banquet municipale (M) Validation
18. Fce-ioi 2.2.10) test13: Choix des emplacements (M) Validation
19. Fce-ioi Tableaux et indices négatifs (C) Cours
20. Fce-ioi Calculer la taille d'un tableau (C) Cours

12 TP12 - FceIOI niveau2.3: Chaînes de caractères

12.1 A - Chaîne Complète

1. Fce-ioi Lire une ligne de texte (C) Cours
2. Fce-ioi 2.3A.1) test1: Petites fiches … (M) Découverte
3. Fce-ioi Comparer 2 chaînes de caractères (C) Cours
4. Fce-ioi 2.3A.2) test2: Priorité alphabétique (M) Découverte
5. Fce-ioi 2.3A.3) test3: 1 ligne sur 2 (C) Validation

En parcourant de vieux livres, vous tombez sur un certain nombre de textes, anodins au premier regard. À y regarder de plus près, ils prennent un tout autre sens, une fois qu'on ne lit qu'une ligne sur deux (la première, la troisième, la cinquième....).
Vous décidez d'écrire un programme permettant d'extraire uniquement les lignes impaires d'un texte donné.

Contraintes

Chaque ligne de texte contient au plus 100 caractères.

Entrée

Sur la première ligne un entier, *nbLignes* : le nombre total de lignes du texte.
Les *nbLignes* lignes suivantes contiennent alors le texte.

Sortie

Vous devez afficher uniquement les lignes impaires.

Exemple d'entrée :

```
13
Mon assistant programmeur est toujours en train de
travailler a son bureau avec assiduite et diligence, sans jamais
perdre son temps en jasant avec ses collegues. Jamais il ne
refuse de passer du temps pour aider les autres et malgre cela, il
termine ses projets a temps. Tres souvent, il rallonge ses
heures pour terminer son travail, parfois meme en sautant les
pauses cafe. Il est une personne qui n'a absolument aucune
vanite en depit de ses accomplissements remarquables et de sa grande
competence en informatique. C'est le genre d'employe de qui on
parle avec grande estime et respect, le genre de personne dont on ne
peut se passer. Je crois fermement qu'il est pret pour la
promotion qu'il demande, considerant tout ce qu'il nous ap-
porte. L'entreprise en sortira grande gagnante.
```

Sortie :

Mon assistant programmeur est toujours en train de perdre son temps en jasant avec ses collegues. Jamais il ne termine ses projets a temps. Tres souvent, il rallonge ses pauses cafe. Il est une personne qui n'a absolument aucune competence en informatique. C'est le genre d'employe de qui on peut se passer. Je crois fermement qu'il est pret pour la porte. L'entreprise en sortira grande gagnante.

6. Fce-ioi Calculer la longueur d'une chaîne (C) Cours
7. Fce-ioi 2.3A.4) test4: Résumé de livres (M) Découverte
8. Fce-ioi 2.3A.5) test5: Lire ou ne pas lire (M) Validation

Aide à la résolution:

L'exercice consiste à afficher un titre si sa longueur est supérieure aux précédentes. Il faut donc rechercher la longueur du titre le plus long (algo de recherche de maximum) et comparer la longueur du titre rentrée à cette dernière

12.2 B - Mots

1. Fce-ioi Lire un mot individuellement (C) Cours
2. Fce-ioi 2.3B.1) test6: Fiches d'inscription (C) Découverte

Au sein de la bibliothèque municipale, toutes les personnes souhaitant emprunter un livre doivent s'enregistrer en indiquant leurs noms et prénoms sur une fiche individuelle conservée à l'accueil.
L'habitude veut qu'ils écrivent d'abord leur nom puis leur prénom, ce qui permet de classer les fiches par ordre alphabétique et permet de rapidement retrouver la fiche qu'on cherche.
Malheureusement, depuis un mois, dans toutes les nouvelles fiches créées les personnes ont indiqué en premier leur prénom puis leur nom !
Votre travail consiste à lire ces couples de prénoms et noms et à les afficher dans le bon ordre.

Contraintes

Chaque nom et prénom est au plus de longueur 100 et ne contient pas d'espace.

Entrée

Sur la première ligne, un entier *nbPersonnes* : le nombre total de personnes concernées.
Sur chacune des *nbPersonnes* suivantes, un prénom et un nom, séparés par une espace.

Sortie

Pour chaque personne, vous devez écrire sur la même ligne son nom, puis son prénom, séparés par une espace.

Exemple d'entrée :

```
4
Alan Turing
Ada Lovelace
Donald Knuth
Claude Shannon
```

Sortie :

```
Turing Alan
Lovelace Ada
Knuth Donald
Shannon Claude
```

3. Fce-ioi 2.3B.2) test7: Analyse de fréquence (M) Validation

En étudiant différents types de textes (romans, lois, article de journaux…), on se rend compte que non seulement les mots utilisés ne sont pas les mêmes mais aussi que leurs longueurs sont statistiquement différentes : par exemple, il est beaucoup plus fréquent de trouver de longs mots complexes dans un article de loi que dans un livre pour enfants.
Afin d'essayer de déterminer automatiquement à quelle catégorie appartient un livre, on souhaite déterminer le nombre de mots de 1 lettre, 2 lettres, 3 lettres… qu'il contient.

Contraintes

Le texte contient un ensemble de mots, séparés par des espaces, sans aucun signe de ponctuation.
Chaque mot contient au plus 100 caractères.

Entrée

La première ligne contient deux entiers : *nbLignes* et *nbMots*.
Chacune des *nbLignes* lignes suivantes contient *nbMots* mots.

Sortie

Pour chaque longueur de mot possible, et uniquement s'il y avait des mots de cette longueur dans le texte, vous devez afficher sur une ligne la longueur et le nombre de mots de cette longueur, séparés par un deux-points (il faut mettre un espace de chaque côté du deux-points).

Exemple d'entrée :

```
2 7
Qui vole un oeuf vole un boeuf
Une abeille vaut mieux que mille mouches
```

Sortie :

```
2 : 2
3 : 3
4 : 4
5 : 3
7 : 2
```

Aide à la résolution:

Tu peux stocker la taille des mots dans un tableau que tu déclares ainsi:

```
nbLettre=[0]*101
```

L'indice correspond à la taille du mot (entre 1 et 100), le contenu correspond au compteur (ie le nombre de fois que les mots de même taille ont été recontrés: il est initialisé à 0)

12.3 C - Caractères

1. **Fce-ioi Accéder aux caractères d'une chaîne** (C) Cours
2. **Fce-ioi 2.3C.1) test8: Impression d'étiquettes** (C) Découverte

Les sous-sols de la bibliothèque municipale sont remplis de milliers de cartons d'archives. Afin d'éviter de passer leurs journées avec la tête tournée à 90 degrés pour pouvoir lire ce qui est écrit sur ces cartons, les bibliothécaires ont adopté un système d'étiquettes où les mots sont écrits de haut en bas avec une seule lettre par ligne.
Étant donné un texte écrit normalement, sur une seule ligne, vous devez afficher l'étiquette correspondante, avec un seul caractère par ligne.

Contraintes

La ligne de texte contiendra toujours moins de 50 caractères.

Entrée

Une seule ligne de texte.

Sortie

Les caractères du texte, affichés verticalement.

Exemple d'entrée :

```
Don Quichotte
```

Sortie :

```
D
o
n

Q
u
i
c
h
o
t
t
e
```

3. Fce-ioi 2.3C.2) test9: Ecriture en miroir (C) Validation

Alors que vous parcourez de très vieux livres, à la recherche d'indications sur le livre qui vous intéresse en particulier, vous tombez sur un langage que vous ne connaissez pas !
À y regarder de plus près, il s'agit des mêmes mots que ceux que vous utilisez tous les jours, mais tout le texte est écrit "en miroir" : toutes les lettres sont écrites de droite à gauche.
Bien que vous arriviez à déchiffrer les textes présents dans ces livres, cela vous prend beaucoup de temps et vous fatigue beaucoup. Vous décidez d'écrire un programme pour remettre automatiquement dans l'ordre les textes.

Contraintes

Chaque ligne de texte contient moins de 1000 caractères.

Entrée

Sur la première ligne, un entier *nbLignes*, le nombre de lignes du texte.
Les *nbLignes* suivantes contiennent chacune une ligne de texte qu'il faut inverser.

Sortie

Pour chaque ligne du texte original, vous devez l'afficher de manière inversée.

Exemple d'entrée :

```
2
tniop a ritrap tuaf li riruoc ed tres en neiR
egangiomet nu tnos ne eutroT al te erveiL eL
```

Sortie :

```
Rien ne sert de courir il faut partir a point
Le Lievre et la Tortue en sont un temoignage
```

4. Fce-ioi Comparer 2 caractères (C) Cours

5. Fce-ioi 2.3C.3) test10: Inscription d'étudiants (M) Découverte

6. Fce-ioi 2.3C.4) test11: ngms sns vlls (C) Entraînement

Les personnes travaillant à la bibliothèque aiment particulièrement se poser des petites énigmes, d'inspiration littéraires. Cette fois-ci, Agrarelle a décidé de créer des énigmes basées sur des titres de livres : elle va supprimer l'ensemble des voyelles (et les espaces) d'un titre et du nom de son auteur et ses collègues devront retrouver le titre original (ainsi que le nom de l'auteur).

Contraintes

Le titre et le nom de l'auteur font chacun moins de 100 caractères.
Ils ne contiennent que des lettres majuscules et des espaces.

Entrée

Sur la première ligne, le titre du livre.
Sur la seconde ligne, le nom de l'auteur.

Sortie

Sur la première ligne, le titre du livre, sans aucune voyelle, ni espace.
Sur la seconde ligne, le nom de l'auteur, sans aucune voyelle, ni espace.

Exemple d'entrée :

```
AUTANT EN EMPORTE LE VENT
MARGARET MITCHELL
```

Sortie :

```
TNTNMPRTLVNT
MRGRTMTCHLL
```

7. Fce-ioi 2.3C.5) test12: La bataille (M) Validation

8. Fce-ioi Déclarer et lire un caractère (C) Cours

9. Fce-ioi 2.3C.6) test13: Analyse d'une langue (C) Validation

Au cours des siècles une langue évolue, et non seulement les mots apparaissent ou disparaissent, mais certaines lettres deviennent plus utilisées ou au contraire moins utilisées. Afin de pouvoir rapidement analyser de nombreux textes, on souhaite mettre au point un programme calculant combien de fois une lettre donnée est présente au sein d'un texte.

Contraintes

Chaque ligne de texte contient au plus 1000 caractères.

Entrée

Sur la première ligne, la lettre majuscule dont on doit chercher le nombre d'apparition dans le texte.
Sur la seconde ligne, un entier *nbLignes* le nombre de lignes du texte.
Sur les *nbLignes* lignes suivantes, le texte, ne contenant aucune lettre minuscule.

Sortie

Un seul entier, le nombre d'apparitions de la lettre au sein du texte.

Exemple d'entrée :

```
E
2
JE VOUS REMECTZ A LA GRANDE CHRONICQUE PANTAGRUELINE
RECONGNOISTRE LA GENEALOGIE ET ANTIQUITE DONT NOUS EST VENU GARGANTUA
```

Sortie :

```
16
```

10. Fce-ioi Modifier une chaîne existante (C) Cours

11. Fce-ioi 2.3C.7) test14: Sans espace (C) Validation

Ecrivez un programme qui lit une ligne tapée au clavier et l'affiche en remplaçant tous les espaces par le caractère "_".

Contraintes

La ligne de texte contient au plus 100 caractères.

Exemple d'entrée :

```
Voici un exemple de texte avec des espaces.
```

Sortie :

```
Voici_un_exemple_de_texte_avec_des_espaces.
```

Contents