1 Turtle Power

All Code Clubs must be registered. Registered clubs appear on the map at codeclubworld.org - if your club is not on the map then visit jumpto.cc/18CpLPy to find out what to do.

Introduction :

Ce trimestre nous allons apprendre un langage de programmation appelé Python. La personne qui a créé Python lui a donné le nom de son émission de télévision préférée : le Monty Python's Flying Circus. Python est utilisé par beaucoup de programmeurs pour beaucoup de projets différents. Python est un langage puissant et intelligent, utilisé par YouTube, la NASA, le CERN et bien d'autres. Si ton club a un Raspberry Pi, tu peux utiliser Python pour le programmer. Beaucoup de personnes aiment utiliser Python parce que c'est un langage qui est facile à lire (comparé à d'autres langages de programmation). Savoir lire du code est très important, aussi important que de savoir en écrire.

Activity Checklist	Follow these INSTRUCTIONS one by one
Test your Project	Click on the green flag to TEST your code
Save your Project	Make sure to SAVE your work now

These projects are for use outside the UK only. More information is available on our website at **www.codeclubworld.org**.
This coursework is developed in the open on GitHub, at **www.github.com/CodeClub**. Come and join us!

Étape 0 : Ouvre l'éditeur de code Python

Si Python est déjà installé sur ton ordinateur, il est temps de commencer.

- [ ] Sous Windows, cherche IDLE dans le menu Démarrer.
- [ ] Sous Mac OS X, ouvre Terminal.app et tape `idle` et appuie sur la touche Entrée.
- [ ] Sous Linux, ouvre un Terminal, tape `idle` et appuie sur la touche Entrée.

Si tu n'as pas Python sur ton ordinateur, pas de panique ! Tu peux télécharger la dernière version de Python sur http://www.python.org/. La version exacte n'est pas importante, mais nous utilisons Python 3, donc tu devrais commencer avec la version 3 (pas la version 2).

Quand IDLE démarre, tu verras une fenêtre qui s'appelle `Python Shell`. C'est dans cette fenêtre que le code s'exécute. Nous allons ouvrir une nouvelle fenêtre pour y écrire notre code. Clique sur `File -> New Window` pour être prêt pour l'étape 1. Assure toi d'avoir les deux fenêtres bien visibles.

Étape 1 : Bonjour tout le monde !

Check-list

- Ouvre IDLE, l'éditeur fourni avec Python. Tout notre code sera écrit dans cet éditeur. Quand tu ouvres l'éditeur, tu verras une fenêtre `Python Shell`, où les résultats et les erreurs de ton programme s'afficheront.

- Si tu ne l'as pas encore fait, clique sur `File -> New Window`. Une nouvelle fenêtre vide s'affichera, avec '*Untitled*' dans la barre de titre.
  Tu devrais avoir deux fenêtres ouvertes maintenant, une pour écrire

These projects are for use outside the UK only. More information is available on our website at **www.codeclubworld.org**.
This coursework is developed in the open on GitHub, at **www.github.com/CodeClub**. Come and join us!

ton programme, et l'autre pour afficher la sortie de ton programme. Sois sûr d'écrire dans la bonne !

- Tape le code suivant dans la nouvelle fenêtre :

```
print("Bonjour tout le monde !")
```

- Vérifie que tu ne sois pas dans la fenêtre appelée `Python Shell`, et enregistre ton code.

Pour enregistrer ton code, clique sur `File -> Save`. Quand l'éditeur te demande un nom de fichier, tape `hello.py`, et enregistre le fichier sur ton Bureau. Clique ensuite sur `Run -> Run Module` pour lancer ton programme.

Félicitations ! Tu viens d'écrire ton premier programme en Python :D (PS : Tu peux lui dire d'afficher tout ce que tu veux, pourquoi ne pas changer ton programme pour lui faire dire bonjour à toi directement ? Change ton programme pour lui faire dire ton nom)

ProTip :

1. Sous Windows et Ubuntu, tu peux utiliser le raccourci Ctrl-N pour ouvrir une nouvelle fenêtre, et ctrl-S pour enregistrer ton fichier après avoir fini de l'éditer, et tu peux utiliser F5 pour lancer ton programme. Sur certains ordinateurs, tu devras peut-être appuyer sur la touche Fn en même temps.
2. Sous Mac OS X, cmd-N pour créer une nouvelle fenêtre, Command-S pour enregistrer ton fichier, et appuie sur la touche fonction (fn) et F5 en même temps pour lancer ton programme.

These projects are for use outside the UK only. More information is available on our website at **www.codeclubworld.org**.
This coursework is developed in the open on GitHub, at **www.github.com/CodeClub**. Come and join us!

Étape 2 : Bonjour, Tortue!

Maintenant, nous allons jouer avec des tortues. Une tortue est un petit robot qui dessine sur ton écran, et on peut lui dire de se déplacer en utilisant des commandes en Python.

Check-list

- Ouvre une nouvelle fenêtre de code, et écrit le code suivant :

```
from turtle import *

forward(100)
```

- Enregistre ton programme dans le fichier `myturtle.py` et clique sur `Run -> Run Module`. Regarde la tortue, est-ce qu'elle a avancé de 100 pixels ? La tortue a un crayon attaché, et trace des lignes quand elle se déplace sur l'écran.

  ProTip: Les fichiers contenant des programmes en Python doivent toujours avoir un nom qui termine par '.py'.

- Essayons de faire bouger la tortue sur notre canevas ! Essaye d'utiliser `backward(distance)`, et aussi de tourner la tortue en utilisant `right(angle)` et `left(angle)`. Par exemple, `backward(20)` dit à la tortue de reculer de 20 pixels, et `right(90)` dit à la tortue de tourner vers la droite de 90 degrés. Tu peux donner plus d'une instruction à la fois, elles seront exécutées dans l'ordre.

```
from turtle import *

speed(11)
shape("turtle")

forward(100)
right(120)
```


These projects are for use outside the UK only. More information is available on our website at **www.codeclubworld.org**.
This coursework is developed in the open on GitHub, at **www.github.com/CodeClub**. Come and join us!

```
forward(100)
left(90)
backward(100)
left(90)
forward(50)
```

Angles et degrés

Essaye un petit peu de dessiner tes propres formes sur le canevas, en utilisant `forward`, `backward`, `left`, `right`. Rappelle-toi, `forward` et `backward` déplacent la tortue en pixels, mais `left` et `right` la tournent en angles. Observons une tortue qui tourne vers la droite.

```
        Nord
          0
          |
Ouest     |      Est
 270 -----+----- 90
          |
          |
         180
         Sud
```

Quand la tortue est tournée vers le Nord, en la faisant tourner de 90 degrés vers la droite, elle est maintenant tournée vers l'Est. La faire tourner de 180 degrés la tourne vers le Sud, et en la faisant tourner de 270 degrés, elle est maintenant tournée vers l'Ouest. Si tu la fais tourner de 360 degrés, elle reste dans la même direction.

Que se passes-t-il quand on tourne vers la gauche?

```
        Nord
          0
          |
```


These projects are for use outside the UK only. More information is available on our website at **www.codeclubworld.org**.
This coursework is developed in the open on GitHub, at **www.github.com/CodeClub**. Come and join us!

```
Ouest     |      Est
 270 -----+----- 90
          |
          |
         180
         Sud
```

Quand la tortue est tournée vers le Nord, tourner de 90 degrés la fait tourner vers l'Ouest, tourner de 180 degrés la fait tourner vers le Sud, et tourner de 270 degrés la fait touner vers l'Est. Si tu la fais tourner de 360 degrés, elle reste dans la même direction. Un tour complet est toujours 360 degrés.

À quoi sert le code au tout début de notre programme ?

1. `from turtle import *` indique à Python que nous souhaitons utiliser la bibliothèque `turtle`, un ensemble de code que nous pouvons ensuite utiliser pour dessiner sur l'écran. Utiliser une bibliothèque nous permet de gagner du temps. ☐
2. `speed()` change la vitesse à laquelle la tortue se déplace. Cette instruction prend une valeur entre 1 et 11. 11 est le plus rapide, 1 le plus lent. ☐
3. `shape()` change la forme de la tortue. Nous utilisons la forme "turtle" pour dessiner une tortue, mais nous pourrions aussi utiliser les valeurs "arrow" (flèche), "circle" (cercle), "square" (carré), "triangle" (triangle) or "classic" (classique). ☐

Nous allons écrire ces instructions au début de tous nos programmes pour le reste de cette leçon. Si tu veux, tu peux essayer de changer la

These projects are for use outside the UK only. More information is available on our website at **www.codeclubworld.org**.
This coursework is developed in the open on GitHub, at **www.github.com/CodeClub**. Come and join us!

forme de ta tortue, ou la faire aller plus ou moins vite.

Étape 3: Dessiner des formes !

Essayons de dessiner un carré en disant à la tortue comment se déplacer.

Check-list

- Ouvre un nouveau fichier dans IDLE, et tape-y le code suivant:

```
from turtle import *

speed(11)
shape("turtle")

forward(100)
right(90)
forward(100)
right(90)
forward(100)
right(90)
forward(100)
right(90)
```

Enregistre ton programme et clique sur `Run -> Run Module`. Est-ce que tu peux voir un carré ? La tortue tourne de 90 degrés quatre fois, et termine dans la même direction qu'au début. En tournant 90 degrés, puis 90 degrés, puis 90 degrés, et encore 90 degrés, la tortue tourne de 360 degrés au total.

Et si on essayait un triangle? Un triangle a trois angles, donc il faut faire tourner la tortue trois fois. Si l'on veut finir dans la même direction, nous allons devoir tourner un total de 360 degrés, comme pour le carré. Donc nous allons tourner de 120 degrés, puis encore de

These projects are for use outside the UK only. More information is available on our website at **www.codeclubworld.org**.
This coursework is developed in the open on GitHub, at **www.github.com/CodeClub**. Come and join us!

120 degrés, et une fois de plus.

- Change ton code pour qu'il ressemble au suivant, pour dessiner un triangle:

```
from turtle import *

speed(11)
shape("turtle")

forward(100)
right(120)
forward(100)
right(120)
forward(100)
right(120)
```

- Exécute ton code. Est-ce que tu as un triangle?

Choisis une couleur

Quelle est ta couleur préférée? Tu peux changer la couleur des lignes en utilisant la fonction `pencolor` (Python est écrit en anglais), et tu peux changer la taille du crayon en utilisant `pensize`:

- Change le code de l'exercice précédent pour qu'il soit identique au code suivant, en ajoutant ces nouvelles commandes:

```
from turtle import *

speed(11)
shape("turtle")

pensize(10)
pencolor("red")
forward(100)
```


These projects are for use outside the UK only. More information is available on our website at **www.codeclubworld.org**.
This coursework is developed in the open on GitHub, at **www.github.com/CodeClub**. Come and join us!

```
right(120)
pencolor("blue")
forward(100)
right(120)
pencolor("green")
forward(100)
right(120)
```

- Lance ton programme, qu'est ce qu'il dessine à l'écran?
  Ce code dessine un triangle avec une ligne épaisse, et en utilisant trois couleurs différentes.

- Essaye de changer les couleurs dans ton code, exécute ton programme, et observe ce qu'il se passe.
  La tortue connait des centaines de couleurs différentes, pas uniquement le rouge, bleu et vert. Essaye d'utiliser ta couleur préférée!
  Tu peux aussi indiquer les couleurs en hex comme nous le faisions en CSS. Au lieu d'utiliser `pencolor("red")`, tu peux te servir de `pencolor("#FF0000")`.
  De quelle couleur est #FF4F00?

Étape 4: Répéter un morceau de code (avec une boucle for)

Le dernier programme que nous avons écrit était écrit avec les mêmes commandes encore et encore. Au lieu de toutes les écrire plusieurs fois, demandons à l'ordinateur de les répéter pour nous. Tu dois avoir déjà vu des *itérations* avec Scratch, en utilisant les blocs `Répéter indéfiniment`, `Répéter x fois`, et `Répéter jusqu'à`. En Python, les boucles for sont utilisées quand tu as un morceau de code que tu souhaites répéter n fois. Dans notre cas, nous souhaitons répéter le code quatre fois (parce

These projects are for use outside the UK only. More information is available on our website at **www.codeclubworld.org**.
This coursework is developed in the open on GitHub, at **www.github.com/CodeClub**. Come and join us!

qu'un carré a 4 côtés).

Check-list

- Ouvre un nouveau fichier et tape le code suivant à l'intérieur:

```
from turtle import *

speed(11)
shape("turtle")

for count in range(4):
    forward(100)
    right(90)
```

- Enregistre ton programme et clique sur: `Run -> Run module`.

  Remarque que le programme est *indenté*, ou décalé vers la droite sous la boucle `for`. Python se sert des espaces afin de savoir quelles commandes il doit répéter. Tu peux utiliser la touche Tabulation pour ajouter des tabulations, et le raccourci Majuscule-Tabulation pour en enlever.

- Essayons de voir ce qu'il se passe si l'on indente que `forward`, et change ton programme pour qu'il ressemble au suivant:

```
from turtle import *

speed(11)
shape("turtle")

for count in range(4):
    forward(100)
right(90)
```

- Tu as remarqué que `forward` est indenté, mais que `right` ne l'est pas

These projects are for use outside the UK only. More information is available on our website at **www.codeclubworld.org**.
This coursework is developed in the open on GitHub, at **www.github.com/CodeClub**. Come and join us!

? Que fait ce programme, à ton avis ? Essaye de l'exécuter pour savoir ?

Est-ce que tu obtiens une ligne droite ? Python répète `forward` quatre fois, et seulement *ensuite* tourne vers la droite. Python se sert des espaces pour grouper des commandes ensemble, de la même manière que Scratch utilise des blocs. Python se plaint si le code n'est pas aligné correctement.

- Change le programme pour revenir a la version précédente, pour vérifier qu'il dessine un carré, mais au lieu d'utiliser des nombres dans le code, nous pouvons utiliser des noms. Cela rend le programme plus facile à comprendre, et cela permet aussi d'arrêter de répéter le même code..

  Change ton fichier pour qu'il ressemble à ceci:

```
from turtle import *

speed(11)
shape("turtle")

sides = 4
length = 100
angle = 90
for count in range(sides):
    forward(length)
    right(angle)
```

- Enregistre ton programme et clique sur `Run -> Run module`.

These projects are for use outside the UK only. More information is available on our website at **www.codeclubworld.org**.
This coursework is developed in the open on GitHub, at **www.github.com/CodeClub**. Come and join us!

Défi: Dessiner les autres formes

Est-ce que tu peux dessiner les formes suivantes en changeant les valeurs de `length` et `angle` ?

- [ ] un triangle? (trois côtés)
- [ ] un pentagone? (cinq côtés)
- [ ] un hexagone? (six côtés)
- [ ] un octogone? (huit côtés)

Souviens-toi, un triangle à trois côtés, et tourne de 120 degrés. Tourner de 120 degrés pour chaque angle veut dire que nous tournons de 360 degrés au total. Pour un carré, nous tournons de 90 degrés pour quatre angles, ce qui nous donne aussi 360 degrés.

Si tu tournes six fois, de combien est-ce que tu dois tourner à chaque fois pour que cela nous donne un total de 360 degrés ? Essaye différents nombres et regarde quelles formes tu obtiens.

Étape 5: Tourne, Tourne, Tourne

Au lieu de devoir calculer l'angle nous-même, pourquoi ne pas demander à l'ordinateur de le faire pour nous. Python est capable de faire certaines opérations sur des nombres, comme les additions, soustractions, multiplications et divisions.

Nous pouvons écrire `sides = 4 + 1` au lieu de 5, ou `sides = 4 - 1` au lieu de 3. Pour une multiplication, Python utilises `*`, et pour une division, on écrit `/`.

S'il faut tourner de 360 degrés au total, on peut calculer l'angle dont on a besoin. Pour un carré, `360 / 4` donne 90, pour un triangle, `360/3` donne 120.

These projects are for use outside the UK only. More information is available on our website at **www.codeclubworld.org**.
This coursework is developed in the open on GitHub, at **www.github.com/CodeClub**. Come and join us!

Check-list

- Modifie ton programme pour calculer l'angle de la manière suivante:

```
from turtle import *

speed(11)
shape("turtle")

sides = 4
length = 20

angle = 360/sides
for count in range(sides):
    forward(length)
    right(angle)
```

- Maintenant change le nombre de côtés. Est-ce que Python dessine les formes correctement ? Essaye de dessiner un hexagone (six côtés), ou n'importe quel autre nombre de côtés !

Étape 6: Forme pleine

Check-list

- Nous pouvons demander à la tortue de remplir nos formes avec une couleur, en utilisant `begin_fill()` et `end_fill()`. Change ton code pour
ajouter ces commandes:

```
from turtle import *

speed(11)
shape("turtle")
```


These projects are for use outside the UK only. More information is available on our website at **www.codeclubworld.org**.
This coursework is developed in the open on GitHub, at **www.github.com/CodeClub**. Come and join us!

```
sides = 4
length = 20

fillcolor('red')
pencolor('red')
begin_fill()

angle = 360/sides
for count in range(sides):
    forward(length)
    right(angle)
end_fill()
```

Comme avec `pencolor`, `fillcolor` change la couleur que la tortue utilise pour remplir les formes qu'elle dessine. Ce programme dessine un carré rouge avec une bordure rouge.

Il faut utiliser `begin_fill()` pour dire à la tortue que tu commences à dessiner une forme à remplir, et `end_fill()` pour lui dire quand tu as terminé.

- Essaye de changer les couleurs, le nombre de côtés, et la longueur des côtés pour voir quelles formes tu peux dessiner !

Étape 7: Le crayon monte, le crayon descend

Check-list

Si tu veux faire bouger la tortue sans laisser de marque derrière elle, tu peux utiliser `penup()` et `pendown()` pour activer ou désactiver le crayon.

- Dans un nouveau fichier, essaye le code suivant:

```
from turtle import *
```


These projects are for use outside the UK only. More information is available on our website at **www.codeclubworld.org**.
This coursework is developed in the open on GitHub, at **www.github.com/CodeClub**. Come and join us!

```
speed(11)
shape("turtle")

pencolor('red')

for count in range(20):
    penup()
    forward(10)
    pendown()
    forward(20)
```

- Ce programme devrait dessiner une ligne en pointillés à l'écran. Lance ton programme et regarde !

Maison, point de départ à l'écran.

Un dernière petite astuce. `home()` ramène la tortue a sa position de départ. `clear()` efface l'écran, et `reset()` ramène la tortue et efface l'écran.

Étape 8: Lâche-toi !

Tu peux utiliser `forward()`, `backward()`, `left()`, `right()`, tu peux répéter des commandes avec `for count in range(4)`, changer les couleurs, changer la vitesse et même remplir des formes avec une couleur !

Est-ce que tu peux dessiner une maison, un oiseau ? Un serpent ? Un chat ? Un chien ? Un lion ? Assemble différentes formes, et vois ce que tu peux dessiner. Est-ce que tu peux dessiner un robot ?

These projects are for use outside the UK only. More information is available on our website at **www.codeclubworld.org**.
This coursework is developed in the open on GitHub, at **www.github.com/CodeClub**. Come and join us!